1963

# Programming a remote monitor and display system for a time sharing computer.

Griffith, Webster.

Monterey, California: U.S. Naval Postgraduate School

http://hdl.handle.net/10945/12900

# PROGRAMMING A REMOTE MONITOR AND DISPLAY SYSTEM FOR A TIME SHARING COMPUTER

WEBSTER GRIFFITH

# PROGRAMMING
# A REMOTE MONITOR AND DISPLAY SYSTEM
# FOR A
# TIME SHARING COMPUTER

* * * * *

Webster Griffith

# PROGRAMMING
# A REMOTE MONITOR AND DISPLAY SYSTEM
# FOR A
# TIME SHARING COMPUTER

by

Webster Griffith

Submitted in partial fulfillment of
the requirements for the degree of

MASTER OF SCIENCE
IN
ENGINEERING ELECTRONICS

United States Naval Postgraduate School
Monterey, California

1 9 6 3

# PROGRAMMING
# A REMOTE MONITOR AND DISPLAY SYSTEM
# FOR A
# TIME SHARING COMPUTER

by

Webster Griffith
Lieutenant, United States Navy

This work is accepted as fulfilling
the thesis requirements for the degree of
MASTER OF SCIENCE
IN
ENGINEERING ELECTRONICS
from the
United States Naval Postgraduate School

# ABSTRACT

The optimization of computer usage from both the users and the computer point of view is discussed. A system wherein several users simultaneously operate separate consoles is explained and specific requirements are laid down for a monitor and display system for a time sharing computer patterned to fit operations at the U. S. Naval Postgraduate School, Computer Center and particularly the CDC 1604 computer and the DD 65 display unit. A more stringent and less flexible system working on the same principle is described and programmed with explanations and instructions for expansion.

# TABLE OF CONTENTS

| Section | Title | Page |
|---|---|---|
| 1. | Introduction | 1 |
| 2. | System requirements | 3 |
| 3. | Description of the general system | 5 |
| 4. | Functions of the system | 17 |
| 5. | Description of the example system | 20 |
| 6. | Conclusion | 30 |
| 7. | Illustrations | 31-38 |
| Appendix-A MACHINE DD65A | | 39 |

# LIST OF ILLUSTRATIONS

| Figure | | Page |
|---|---|---|
| 1. | Interrupt handler | 31 |
| 2. | Idle loop | 32 |
| 3. | DD65 keyboards | 33 |
| 4. | Flow chart of DD65A | 34 |
| 5. | Flow chart of KYBORD | 35 |
| 6. | Flow chart of MODE | 36 |
| 7. | Flow chart of PICGEN | 37 |
| 8. | Flow chart of PACK | 38 |

## 1. Introduction.

The conflict between the large, fast and flexible digital computer and its slow human master has led to the loss of efficiency by having multi-million dollar computers idle as the human operators think.

One answer, and the subject of this thesis, is to let many users operate the computer simultaneously at separate and remote locations to combat the user inefficiency in the computer installation at the U. S. Naval Postgraduate School.

A control system which would permit the user to inspect, alter or compose a program from a remote console was desired. Available equipment included a CDC 1604 Computer[1] and a DD 65 Display Unit[2], the latter is a device with two cathode ray tubes, character generator, a random access memory for storage (reading is sequential to the character generator and associated logic circuits only), two keyboards for input and the necessary equipment to allow compatability with the CDC 1604, CDC 160 and assorted radars. Direct linkage is not provided between the keyboards, the display and the user.

General specifications for this type of a system are difficult unless exact requirements are prescribed. The example used is therefore a limited case, however it is hoped that it can lead to a larger, fuller system.

---

[1] Control Data Corporation, Computer Division, Control Data 1604 Computer Programming Manual, Publication 167, Undated.

[2] Instruction Manual for the DD 65 Display Unit.

The system should be capable of utilizing as much of the flexibility of the DD 65 as possible and still allow the user an easy to operate console.

Since the "Monitor" system in use at present is more or less adequate for normal stacked job operation, this system is aimed for the user requiring immediate access to his program or to its input or output. This requirement is such that little modification or addition is needed to expand the functions of the system to those mentioned in section 5.

The principle effort was directed to producing a system which would allow symbol manipulation in such a way as to achieve compatability between not only the equipment used but also the equipment and the users.

The example program is designed for use with the FORTRAN 60 Compiler[1] although the general system could handle as many compilers/assemblers as needed.

Besides the control system there are two principle subsystems:

a. A keyboard to intermediate code compiler called KYBORD.

b. A intermediate code to display unit compiler called PICGEN.

Other transcoders are of a more minor nature in design if not in importance.

---

[1] Control Data Corporation, Computer Division, FORTRAN System for the Control Data 1604 Computer, Publication 087A, 1961.

### 2. System Requirements.

There will be two systems considered:

a. The general system.

b. The example system.

The example system is a lesser included part of the general system which is designed to demonstrate the practicability of the general system. The program of the example system and an explanation is given later. An "*" denotes requirements for the example system.

The system requirements are as follows:

a. Computers - CDC 1604 and two CDC 160's.

*- CDC 1604.

b. Consoles - CDC 1604, two CDC 160's and DD 65.

*- CDC 1604 and DD 65.

c. Input/output (on-line) - 9 magnetic tape units, 3 flex punches, one of 3 radar's, experimental devices via a D/A converter, graph plotter, typewriter, DD 65 typewriter, and card reader.

d. Compilers and Assemblers - FORTRAN 60, COOP MONITOR, JOVIAL, NELIAC, SCRAP, AR, and octial machine inputs.

*- FORTRAN 60.

e. Libraries - Any proven programs in the above form.

e. Libraries (Cont'd)*- One integrated program as an example.

f. Delays permitted - Only those delays normally encountered in FORTRAN 60 plus actual run times.

g. Edit features - Insertions, deletions and exchanges can be made on programs on tape, in memory, and on the display tube.

*- No changes in tapes.

h. Location of control - Control is in resident.

*- Control is in Program DD 65A.

3. Description of the general system.

a. Resident program.

Under present conditions at the U. S. Naval Postgraduate School Computer Center, the fastest input/output medium is binary magnetic tape. This is a most significant factor in the operation of the system control.

A single resident and control program is required since the time delay in loading and unloading individual resident programs for individual consoles or special purpose usage would not permit the fulfillment of the requirement of instantaneous (for humans) response to DD 65 keyboard hits. This places a fundamental limitation on the system as a single resident program must be large for flexibility of control and display and small for flexibility of the input programs of the users.

The selection of the exact bias point is a delicate decision and the expedious answer of providing all that might possibly be used or only as much as is presently used is not wise since, if the system is to be put in general use, the users must be told a firm number of memory cells which they have available. If the bias is altered the user's previously checked out programs may either not run due to insufficient room or not run efficiently due to not utilizing all the memory available.

The bias point was calculated as follows:

| | | |
|---|---|---|
| (1) | Basic FORTRAN 60 resident | 5000 cells (octal) |
| (2) | DD 65 processor (KYBORD) | 2400 cells (octal) |

(3) Buffer for DD 65 memory. (one buffer for each DD 65) 1000 cells (octal)

(4) Control system program 400 cells (octal)

Or 10,000 cells plus 1000 for each DD 65; over 1/8th of memory. Any further encroachment on the available memory space might easily limit the usefulness of the whole system and confine its use to only a portion of the user traffic.

b. Simultaneity and interrupt priority.

Since DD 65 keyboard hits must be processed without humanly discernible delay and since interrupting the keyboard in mid-routine requires the storage of many flag and switch settings and, also, since the keyboard routine requires less than one milli-second; interrupts actuated by a DD 65 not at present in control will be deferred if the console in control is processing a keyboard hit which does not directly activate a user program.

The interrupt priority list is as follows:

(1) A "normal" interrupt (e.g. "Interrupt on channel 5 active" or "Interrupt on Arithmetic Faults") will be processed as occurring.

(2) A DD 65 key hit interrupt can interrupt a running program but not another keyboard hit.

(3) A program run initiation can interrupt only when the control program is passive or idling. Once initiated a program can be interrupted by (b) or (a) above but not by another program which must wait until the present program is finished.

Priority Chart

| | Idle | Program Being Run | Keyboard Hit Being Processed | Normal Int. In Process |
|---|---|---|---|---|
| Program to be run | Do | Wait | Wait | Wait |
| Key hit to be processed | Do | Do | Wait | Wait |
| Normal interrupt to be processed | Do | Do | Do | Do |

Table 1

A flow chart illustrating the interrupt scheme is shown in Figures 1 and 2.

c. Keyboard decoder.

The keyboard decoder will analyze the input from the designated input typewriter. It will be noted that each DD 65 unit has two keyboards; Keyboard #1 has a modified and enlarged standard keyboard, Keyboard #2 is a special keyboard which can be programmed as desired and consists of 34 keys each with selectable lights, in addition the meaning of the keys can be altered by using any one of eight overlays which transmits a three bit code along with the six bit code of the individual key. Even with the large number of keyboard #2 keys available, it is still inconvenient to govern all of the possible situations from Keyboard #2 only. Therefore, I have made Keyboard #2 the category selector and have let Keyboard #1 select subcategories as appropriate. For example, if "MODE" is selected by Keyboard #2, then "CHARACTER", "VECTOR", or "SPOT" is selected by hitting "C", "V", or "S" respectively on Keyboard #1.

In general then, the KEYBOARD DECODER analyzes the key hit in light of the key hit history (e.g. A "C" hit after "MODE" has been hit has an entirely different meaning than a "C" hit after "TYPE" has been hit) and sets switches and selects subroutines which will be used later in the KEYBOARD PROCESS portion of the program. The system of category selection on Keyboard #2 and subcategory selection on Keyboard #1 was chosen because:

(1) The number of Keyboard #2 keys under one overlay is limited to 34. Even the limited example system has 48 subcategories plus numerical selects.

(2) If multiple Keyboard #2 hits were used to select subcategories, a problem as to labeling would arise. One can label the category key easily but the subcategory code could not be labeled and would require the user to either remember or make continued reference to a code sheet.

(3) By allowing Keyboard #1 to denote subcategory selections, easily remembered codes can be used (e.g. "S" for small, "R" for right, "P" for pipper, etc.). In addition the number section and sign keys are handy (e.g. a right margin 63 spaces to the right of center is selected as follows: hit "RIGHT MARGIN" key on Keyboard #2 and then hit "+", "6", and "3").

d. Keyboard processor.

Once the KEYBOARD DECODER has completed its operations the KEYBOARD PROCESSOR is called and carries out the execution of the steps required by the particular hits.

e. The intermediate code.

Since information in the DD 65 memory is not retrievable except to be displayed on the DD 65 scopes, some additional storage is required. With our speed requirements only CDC core will serve. This information could be in an of three forms:

(1) Raw input as received in the computer.

(2) Processed input in a form directly transferable to the DD 65 memory.

(3) Processed input of a uniform nature but not necessarily in the form suitable for DD 65 memory.

The first form requires that every type of input be reprocessed by its own peculiar coder/decoder whenever updating or retrieval are required. The second form requires multiple input coders but only one decoder, however the decoder must unpack bits in highly condensed data words and would be slower than the third form which uses a special coder to place each type of input into a simple intermediate code. This code is stored in a buffer area and is identical for all inputs. This is the system used. Each form of input (e.g. DD 65 typewriter, BCD magnetic tape, flex tape) has its own coder which stores the transformed input in an area called PICBUF. Each of these coders are reciprocal and one can go from flex tape to PICBUF and back without loss. However, since all of the input forms differ in some respect, going from the DD 65 typewriter to PICBUF, to magnetic tape, back to PICBUF, and finally to the DD 65 display could, but would not necessarily, involve some loss.

f. Picture generator.

The picture generator (PICGEN) transforms the intermediate coded PICBUF into the form suitable for the DD 65 and transmits this code to the DD 65. There are several ways to approach the use of PICGEN for updating. One, and normally the fastest, is to only process or reprocess the last word sent to the DD 65 memory. This method has the disadvantage of being difficult to edit and some forms of editing are particularly cumbersome. Another method is to update only the last line of display on the DD 65. This allows easy editing of the last line but has the same difficulty encountered in the first method in deeper editing. The method chosen is to update the entire DD 65 memory each time a character is added to PICBUF. This is by far the slowest method but does allow easy and complete editing features. Since PICGEN returns to the control program as soon as an "end of file" symbol is reached, only when PICBUF is nearly full will the time required to decode be of any consequence. The time from entry to exit under worst case conditions is .36 seconds and under normal conditions would be about .10 seconds. For example, a man typing at a rate of 50 words a minute averages .20 seconds per key hit, therefore only a fast typist typing 128 characters per line at the bottom of the page would notice any reduction in his normal typing speed.

# THE PICGEN CODE

00 - Begin string. All bytes after this are transmitted directly to the DD 65 until a 77 byte is used.
01 - Disregard all bytes until this one is processed.
10 - Set for character mode.
11 - Set for vector mode.
12 - Set for spot mode. The next three bytes represent the X (1st 9 bits) and Y (last 9 bits) location of a ".". Exit from the string is automatic. If more than one spot is to be sent, each must be proceeded by a begin string code (00).
20 - Set X location equal to the value of the next byte.
21 - Set Y location equal to the value of the next byte.
22 - Set X location equal to - the value of the next byte.
23 - Set Y location equal to - the value of the next byte.
30 - Set for small size characters.
31 - Set for medium size characters.
32 - Set for large size characters.
40 - Set for display on right tube.
41 - Set for display on left tube.
50 - Set for normal intensity.
51 - Set for bright intensity.
60 - Set for auto-incrementation right.
61 - Set for auto-incrementation down.
76 - End of file, exit PICGEN.
77 - Disregard this byte.

# THE CHARACTER CODE

00 - Blank
01 - 1.
02 - 2.
03 - 3.
04 - 4.
05 - 5.
06 - 6.
07 - 7.
10 - 8.
11 - 9.
12 - 0
13 - =
14 - ≠.
15 - :.
16 - '.
17 - [.
20 - Space
21 - /.
22 - S.
23 - T.
24 - U.
25 - V.
26 - W.
27 - X.
30 - Y.
31 - Z.
32 - ].
33 - ,.
34 - (.
35 - →.
36 - Tab.
37 - ≥.
40 - _.
41 - J.
42 - K.
43 - L.
44 - M.
45 - N.
46 - O.
47 - P.
50 - Q.
51 - R.
52 - }.
53 - $.
54 - $.
55 - ↑.
56 - ||.
57 - ∧.
60 - +.
61 - A.
62 - B.
63 - C.
64 - D.
65 - E.
66 - F.
67 - G.
70 - H.
71 - I.
72 - <.
73 - ..
74 - ).
75 - {.
76 - CR.
77 - End of string.

## THE VECTOR CODE

The vector code describes the size, direction and "on" condition of the vector in six bits as follows:

0X - Blank, small vector.
1X - Blank, large vector.
4X - Unblank, small vector.
5X - Unblank, large vector.
X0 - To the right.
X1 - To the right and up (elevated 45° to the right).
X2 - Up.
X3 - To the left and up (elevated 45° to the left).
X4 - To the left.
X5 - To the left and down (depressed 45° to the left).
X6 - Down.
X7 - To the right and down (depressed 45° to the right).
77 - End of vector string.

## THE SPOT CODE

The spot code consists of 18 bits, 3 bytes or 6 octal numbers. The first 9 bits represents the X location and the last 9 represents the Y location. The tube is divided into 512 (1000 octal) lengths in each direction with the (0,0) location at the center. The location code is in one's complement notation.

Examples: (1) 377000 is right center
(2) 400400 is the lower left corner.
(3) 000000, 777777, 000777, and 777000 are all dead center.

An example: If the first few words in PICBUF looked like this:

```
10 10 31 40 50 60 23 77
21 77 00 62 65 67 71 45
20 70 65 51 65 77 77 76
```

It would mean to start the file (01), use character mode (10) of medium size (31) on the right tube (40) of normal intensity (50) and increment to the right (60). Start at the left (23 77) upper (21 77) corner. String follows (00). "BEGIN HERE". Stop the string (77) and exit (76).

g. Other decoder or transcoders.

Other media must be translateable to and from the PICBUF code. In this case; Flex paper tape, binary magnetic tape, BCD magnetic tape, binary core and BCD core. The actual source of the code is not important to the code since all will be delivered to a buffer area in core and separate routines will load and dump into these buffers. Therefore, we need three decoders and their reciprocals; Flex, BCD, and binary to PICBUF code and PICBUF code to Flex, BCD, and binary in addition of course, DD 65 keyboard to PICBUF and PICBUF to DD 65 display.

(1) Flex to PICBUF.

(a) Upper case to medium size and lower case to small size characters. The reverse in numerals.

(b) Change color of ribbon works a toggle switch on the intensity code.

(c) Characters not present to "$".

(d) "&" to plus sign.

(e) "*" to times sign.

(2) PICBUF to Flex.

The reverse of (1) above except large to upper case and all brackets to "(" and ")". It might be pointed out here that since PICBUF code is the most flexable, care must be taken in forcing some transcriptions like "Increment Down". One can force other media to increment down, however, one can not make them backspace. Impossible transcriptions like "Left Tube" are ignored.

(3) BCD to and from PICBUF

This is straight forward since the FORTRAN 60 BCD characters are relatively few and non-character selects absent.

(4) Binary to and from PICBUF.

The octal, 16 number representation is used.

(5) Vector to and from Graph Plot.

This transcription in either direction is within the capabilities of the system but has not been investigated further.

h. A call to run a user's program.

If the size and complexity of the system are to be retained within bounds, and automatic super-master program which could rebootstrap, control time length of runs under various priority conditions, permit check out program runs, interpretively run non-checked out programs, etc.; can not

be included nor do I feel it would be desirable. Since four users are being kept busy, the expense of one operator would be well worthwhile especially since monitor programs could be done whenever the computer was between runs. Therefore, all programs will be allowed to run if space permits. If a bad program clobbers the system the operator can easily bootstrap and he can cause premature exits from programs which take too long. Here, it might be pointed out is another advantage of the human operator since the users through the intercom can make special requirements known to the operator who can allow or not extra long programs and advise the other users if long delays are expected. The operator could also load special libraries and activate on-line equipment if desired.

4. Functions of the system.

Once the control system is in operation, it can be used for several functions; the diversity of which is determined by the sophistication of the program and the equipment available

a. Student's computer.

This mode of operation is designed for use by one not familiar with all intricacies of computer operation. In fact only previously checked out programs on a specific library will be permitted to run. The only option of the user will be to pick up the desired program and insert data. In most cases the output will be to the DD 65 display with an option for a permanent record when desired. Suitable error exits must be made for input data which is not appropriate to the program and likewise all output data must be tested for its fit on the display and printer (or graph plot).

Consequently only programs of a general and continuing nature would warrant the programing of a problem. As an example an instructor could ask a class studying Fourier analysis to input wave forms and observe the output. The student in this case would place vectors on the display to represent the wave form and a bar graph representation could represent the amplitude of the sin and cos coefficients.

In addition, if the results of a program under many varying parameters is to be studied at great length, this mode could allow untrained observers to procure data with little effort.

b. Check-out computer.

This mode of operation would allow the user to write, edit, compile and run FORTRAN programs.

One often is faced with a long time delay from the completion of coding to the commencement of useful runs due to the time consumed debuging their programs. In the check-out computer, a program to be tested is loaded on a tape unit or fed from the satellite 160. The user then requests from the DD 65 a compilation of the program. If compilation errors result they will appear on the display. The user calls in his program to the display tube, makes the necessary corrections, and reloads the tape unit. The process is repeated until all errors of compilation are eliminated. Then the program is run and all errors in running are likewise corrected. When the program is completely error free the user can ask for an on or off-line punch of new cards containing the revised program.

Previously a user had to either wait for the next monitor run or; if operating the computer personally, the computer was in an idle state for long periods while cards were changed, etc.

This computer is not as simple to use as the students' but would require little more knowledge than is needed for FORTRAN programing.

This computer would also be used for parameter variation problems as in the student's computer but would not

require the rigorous testing and "libing" of the latter.

c. Experimental display computer.

Using this mode, on-line experimental data is fed through the satellite CDC 160 to the CDC 1604 and the processed information is displayed on the DD 65.

This offers a unique opportunity to adjust not only the experimental equipment but also the processor parameters in a rapid sequence which would allow much quicker optimazation of the equipment.

d. Teaching machine.

Although the student's computer is a teaching machine some such devices are of such a special nature as to require individual libraries and would have to be programmed as needed.

e. Learning machine.

As in d. above, special programs would be required.

5. Description of the example program.

As previously mentioned, this program is a lesser included program of the general system designed to show the latters practicability by providing a working system and to form a basis upon which to build the general system.

a. Basic program (DD 65A).

This program is the executive or control program and determines when, what and how soon a subroutine is to be called and what the inputs to the subroutine will be. The Idle Loop and Interrupt Handler would be inserted here in the general system but they are not necessary as such in this limited system. Note that the status reports are sent to DD 65A via the Q-register.

b. Keyboard handler (KYBORD).

This subroutine incorporates both the Keyboard Decoder and the Keyboard Processor of the general system. It has two arguments: 1. The input byte, 6 bits, right justified in the A-register, 2. The keyboard from which it came in the Q-register. The output, also in the Q-register, indicates where the next keyboard hit should originate, however, this is of information value only to the DD 65A since the latter can over-ride any recommendations of KYBORD.

In this and other subroutines routing is used whereby information in one hit alters the processing route of a future hit(s).

The keys are grouped as follows:

(1) Keys which set the characteristics of the

display when Keyboard #1 is subsequently hit.

(a) "Mode"

When the "Mode" key (see figure 6) is hit the routing is adjusted for the following possibilities:

1. "C" is hit next.
2. "V" is hit next.
3. "S" is hit next.
4. A Keyboard #1 key not "C", "V", or "S" is hit next.
5. A Keyboard #2 key is hit next.

If case 5. action is taken in accordance with the new Keyboard #2 hit and the "MODE" hit is ignored. In case 4. an error exit results which displays a flashing "KBI ERROR" on the typing tube and the control program goes into idle. In cases 1., 2. and 3. "CHARACTER", "VECTOR", or "SPOT" is displayed on the status tube and appropriate initiating action is taken in the PICBUF region. At the completion of the action DD 65 awaits the next Keyboard #2 hit.

(b) "SIZE"

The remainder of the characteristic Keyboard #2 keys are processed in a similar fashion to "MODE" and inappropriate hits are handled exactly the same.

Appropriate Keyboard #1 hits are:

1. "S" for "SMALL 126"
2. "M" for "MEDIUM 63"
3. "L" for "LARGE 31"

The indications of size and maximum number of characters per line are displayed on the status tube. In addition and in conjunction with the selection for the margins the actual number of characters per line ("CHAR/LINE XXX") is calculated and displayed on the status tube.

(c) "TUBE"

Appropriate Keyboard #1 hits are:

1. "R" for "RIGHT TUBE".
2. "L" for "LEFT TUBE".

It is well to mention that all of the characteristics selection displays are stored in a separate section from PICBUF and therefore are not included in any dump or decoding of PICBUF.

(d) "INT" (intensity)

Appropriate Keyboard #1 hits are:

1. "N" for "NORMAL".
2. "B" for "BRIGHT".

(e) "INCR" (increment).

1. "R" for "INCR RIGHT".
2. "D" for "INCR DOWN".

(f) "SPACING"

Appropriate Keyboard #1 hits are:

1. "S" for "SINGLE" spacing.
2. "D" for "DOUBLE" spacing.
3. "T" for "TRIPLE" spacing.

(g) "PIP"

The "PIP" key is an on/off switch. When

on the "PIP" key is lighted an "X" appears on the typing tube in the position of the pip. The pip's position is controlled by the four keys "UP", "RIGHT", "DOWN" and "LEFT" which are called direction keys and operate as follows: If the direction key hit is not the same as the last direction key hit, the pip is not moved, but if the direction key is the same the pip is moved in the direction indicated by an amount equal to one space more than it moved the last time. For example if the "UP" key was hit and then the "RIGHT" key was hit ten times in succession, the pip would move as follows:

|  | "RIGHT" hits |  |  |  |  |  |  |  |  |  |
|---|---|---|---|---|---|---|---|---|---|---|
|  | 1st | 2nd | 3rd | 4th | 5th | 6th | 7th | 8th | 9th | 10th |
| Spaces moved to the right | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 |
| Relative pip position | 0 | 1 | 3 | 6 | 10 | 15 | 21 | 28 | 36 | 45 |

The pip has end around action, that is if one continues to hit the "RIGHT" key after the pip is on the right extreme of the tube, the pip will appear on the extreme left of the tube.

(h) "LEFT MARGIN".

Appropriate Keyboard #1 hits.

1. "P" for "PIP" where the left margin is taken from the horizontal position of the pip.

2. "+" or a "-" followed by two octal numbers where the left margin is taken from the number given considering the tube center to be 00 and to extend to

77 (oct) spaces in either direction. An indication of the margins is displayed on the typing tube by an upward pointing arrow and the octal position of the arrow displayed underneath.

(i) "RIGHT MARGIN" (same as left margin).

(j) "TAB SET".

This hit does not wait for a Keyboard #1 hit but uses the horizontal position of the pip as the desired tab setting. A tab is displayed as a caret on the typing tube (top). There are 16 tabs available and until all 16 are used elsewhere a tab will always be set at the right extreme of the tube.

(k) "TAB CLEAR"

The pip is again used as a position reference. If no tab is in the position indicated, the "TAB CLEAR" key will blank thrice and the program will return to idle.

(l) "MARKER"

The marker indicates where the next character will be placed or where the end of the last vector is (an aid in using blank vectors) as appropriate. The particular symbol to be used as a marker is determined by the Keyboard #1 key (including the space bar) immediately after the "MARKER" key is hit.

(m) "START"

The "START" key initially locates the marker and operates as follows:

1. "T" - Start at the top of the tube and at the left margin.

2. "P" - Start at the position indicated by the pip.

3. "Y" - Start at the left margin and vertically as indicated by the pip.

(n) "TRACK BALL"

The "TRACK BALL" key is an alternate way to position the PIP.

(o) General

All of the preceeding keys affect the character mode but in the vector mode all indications are ignored except Mode, Tube, Marker and Start and in the spot only Mode, Tube, Pip, and Start are recognized. However, one may switch from mode to mode without reselecting. For example if Size is set for Large and one then transfers to Vector mode using small vectors, a transfer back to character Mode will result in using Large size characters unless altered.

The light associated with a key will remain on as long as additional information is required from Keyboard #1. A blinking light indicates that the function of the key could not be completed and when the blinking stops control is returned to the executive program.

(2) Set to type

The "TYPE" key activates Keyboard #1 and until another Keyboard #2 key is hit the Keyboard #1 keys will

operate as previously selected.

(a) In the "CHARACTER" mode normal typing procedures are used and, in addition, an automatic carriage return will result when the last character or space which will fit between the margins has been typed. The caret of the marker will always indicate where the next character will go. Tabs operate in a normal fashion except tabing past the left margin will result in an automatic carriage return.

(b) In the "VECTOR" mode only part of the keyboard is activated, that is the numerals 0 through 9 and the letters "S", "M", "L", and "E". The letters and "0" and "5" do not cause a vector to be formed but modify subsequent vectors as follows:

1. "S" - A small vector.
2. "M" - A large vector.
3. "L" - Two large vectors.
4. "E" - Four large vectors.
5. "5" - Unblank vectors.
6. "0" - Blank vectors.

The remaining numerals from vectors and their direction can be determined by the number key's relative position from the "5" key. They are as follows:

1. "1" - Southwest (The top of the tube is North
2. "2" - South
3. "3" - Southeast

4. "4" - West

5. "6" - East

6. "7" - Northwest

7. "8" - North

8. "9" - Northeast

(c) In the "SPOT" mode Keyboard #1 is not activated. While in this mode, a spot is generated each time the "TYPE" key is hit and the position of the spot is taken from the pip.

(2) Edit features

(a) The Keyboard #1 "blank" key acts as a combination backspace and detete.

(b) "CHAR EDIT" locates the character indicated by the pip and will then proceed as follows:

1. If indicated character can not be found the "CHAR EDIT" key will blink and exit the routine.

2. If "D" on Keyboard #1 is hit, the indicated character is deleted and all characters to the right on that line are moved to the left one space.

3. If "R" on Keyboard #1 is hit, the indicated character is deleted and replaced by the next hit on Keyboard #1.

4. If "I" on Keyboard #1 is hit, the next hit on Keyboard #1 is inserted just to the left of the pip. This key operates only in "CHARACTER" mode.

(c) "LINE EDIT" locates the line and character indicated by the pip and proceeds as follows:

1. If the line and character indicated can not be found, the "LINE EDIT" key will blink and exit the routine.

2. If "D" is hit the entire line indicated will be deleted and the following lines moved up to fill in.

3. If "R" is hit the indicated character and all to the right on the indicated line will be deleted. Keyboard #1 hits will then be inserted in sequence starting with the location of the pip and continue until either the end of the line or until "CR" is hit at which time the routine will exit. "TAB" will be ignored.

(d) "RESET" will extinguish all Keyboard #2 lights except its own and end the current string.

(e) "CLEAR" clears PICBUF and prepares it for a refill (starts file). It also clears the displays which are no longer appropriate.

c. Picture generator (PICGEN)

PICGEN is a transcoder which converts the intermediate PICGEN code to the machine code acceptable to the DD 65 display. The PICGEN code is composed of two parts; one of which, the string code, is merely a modified BCD code for transmitting specific characters, vectors or spots. The other code establishes the characteristics of the display. Both codes are in six bit bytes, 8 bytes to a word and words in numerical increasing order starting at address PICBUF+1. The program PICGEN shuffles through PICBUF byte by byte in sequence forming the proper skeletal words for input to the

DD 65 display until the "00" code causes a shift in mode to string processing whence each byte is inserted in the proper location word by word until the "77" code is reached which terminates the string and the original mode of operation is reactivated.

Since all 64 codes in the string code are meaningful there are no error exits in that mode, however, impossible characteristic codes cause an exit and error display on the right tube.

The process is continued until a "76" code is processed at which time an exit from PICGEN is taken. If a "76" code is not processed when PICBUF is exhausted an exit will also occur.

d. Character locator (CHALOC)

CHALOC uses the pip location to find a specific character in PICBUF and relay the information back to the calling program. If no character is found at the designated location, a -1 is inserted in the A-register. This routine could be expanded to trace vectors and spots but at present this cannot be handled nor can the vertical incrementation be processed.

## 6. Conclusion.

A time sharing, remote monitor and display system is feasible with equipment presently available at the U. S. Naval Postgraduate School.

The use of a DD 65 Display Unit in conjunction with a CDC 1604 computer with a FORTRAN 60 compiller permits a user to monitor and view his program and its input and output with greater ease and at the same time not interfere or delay appreciably with the normal operations on the main console.

An intermediate code or language of some sort is necessary to allow compatability of various input and output equipment. Its form will be determined by the specific use to be made of the system and the equipment peculiarities, however, the more general code will allow easier additions and modifications.

# INTERRUPT HANDLER

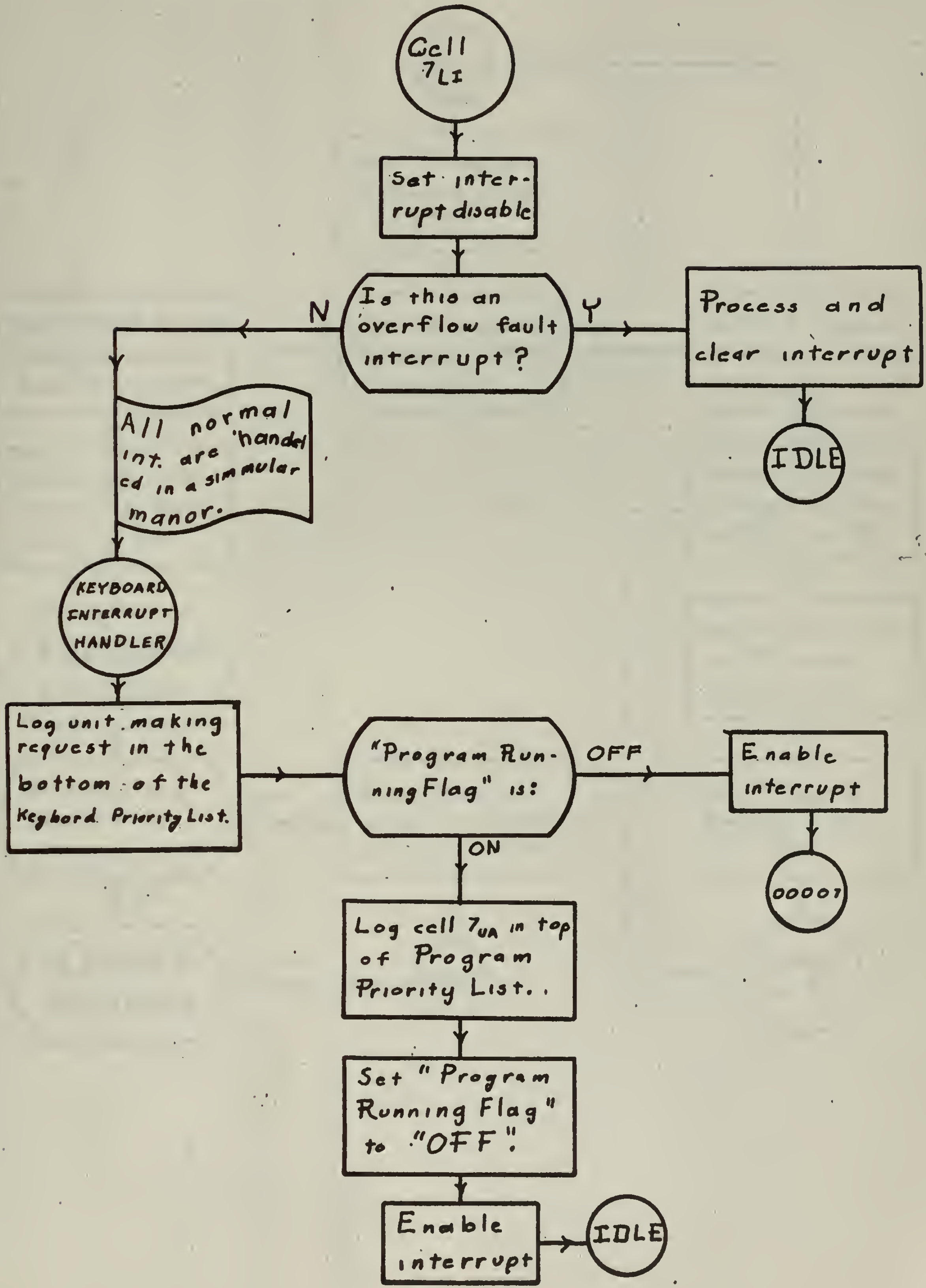


Fig 1

[illegible]

# IDLE LOOP

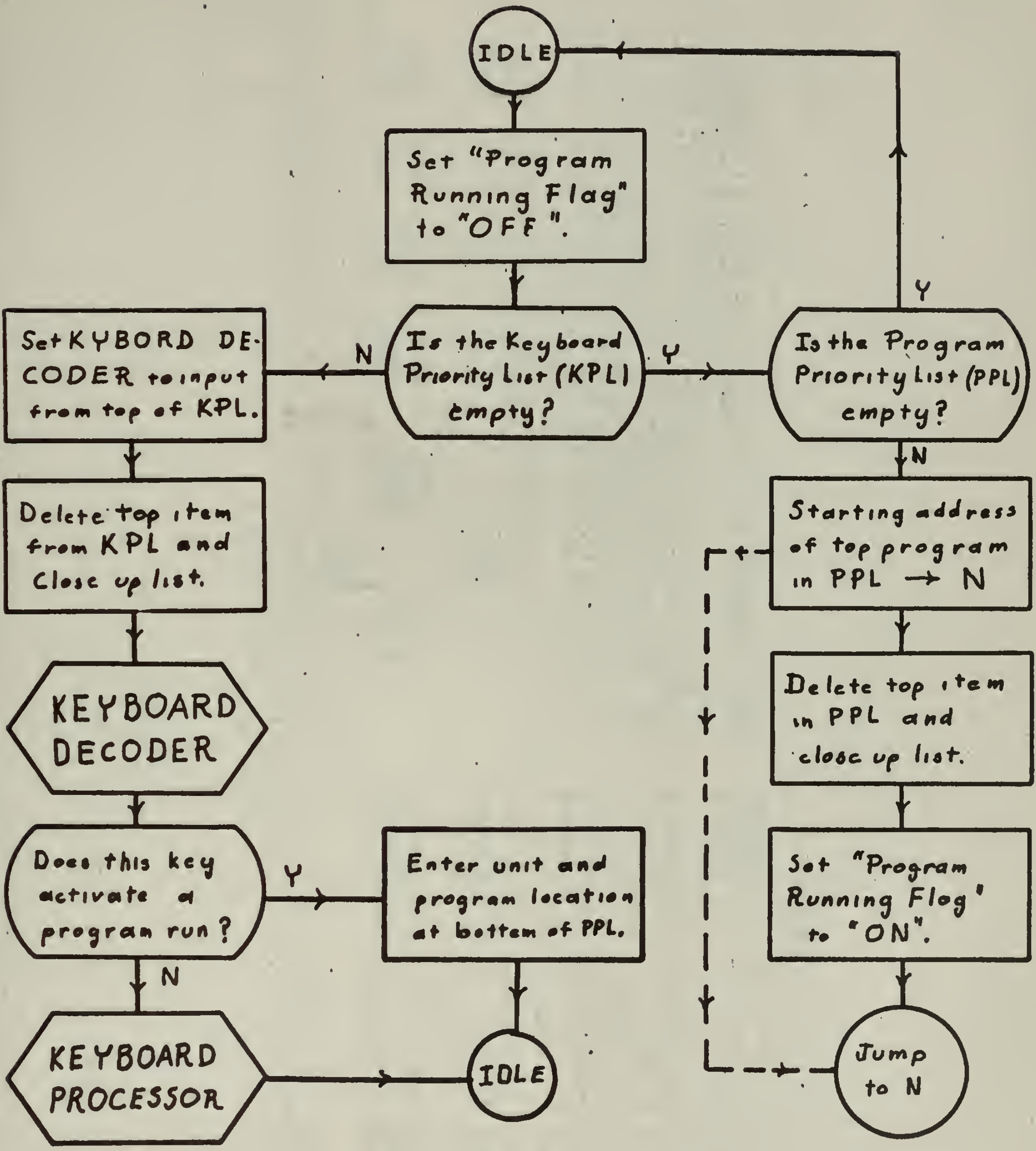
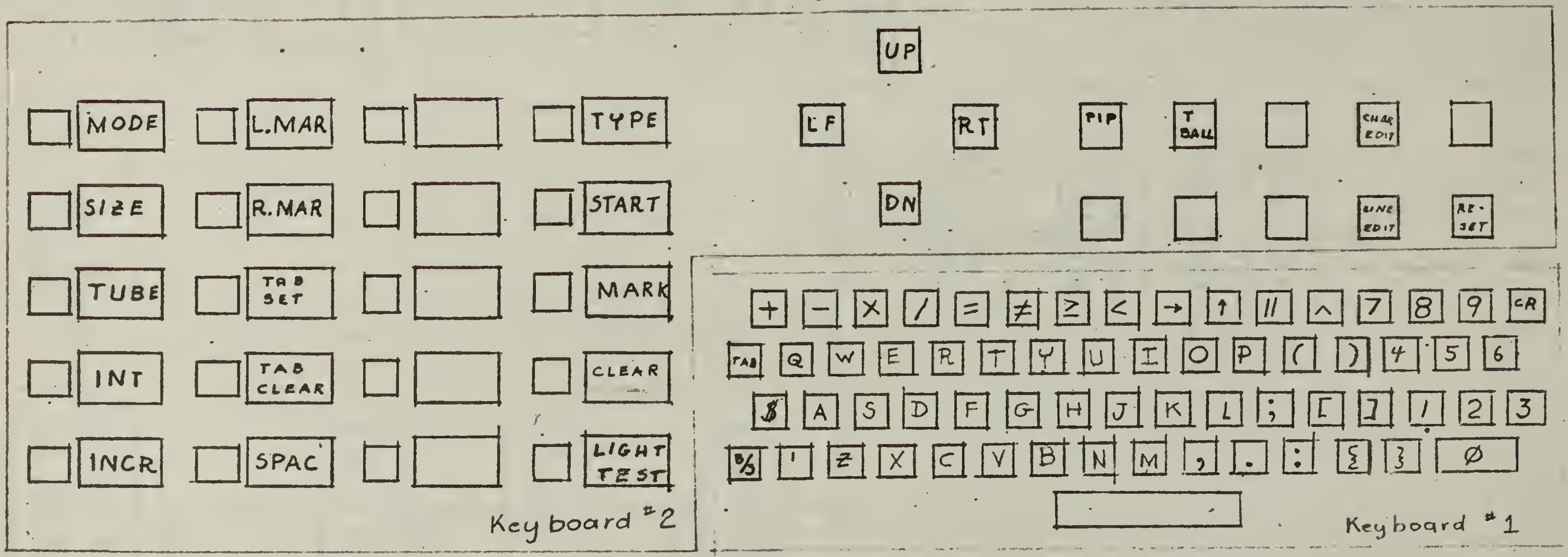
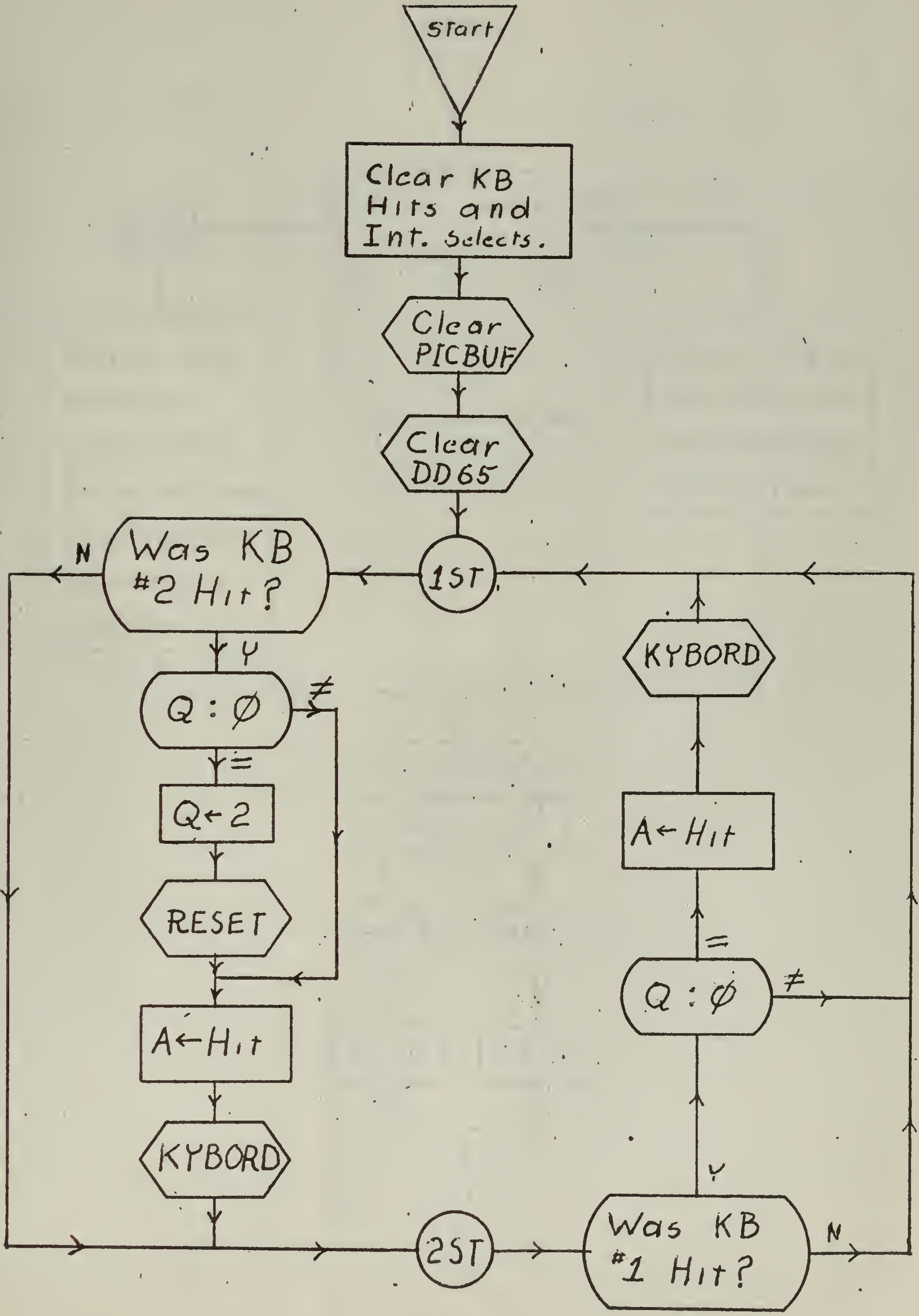


Fig 2

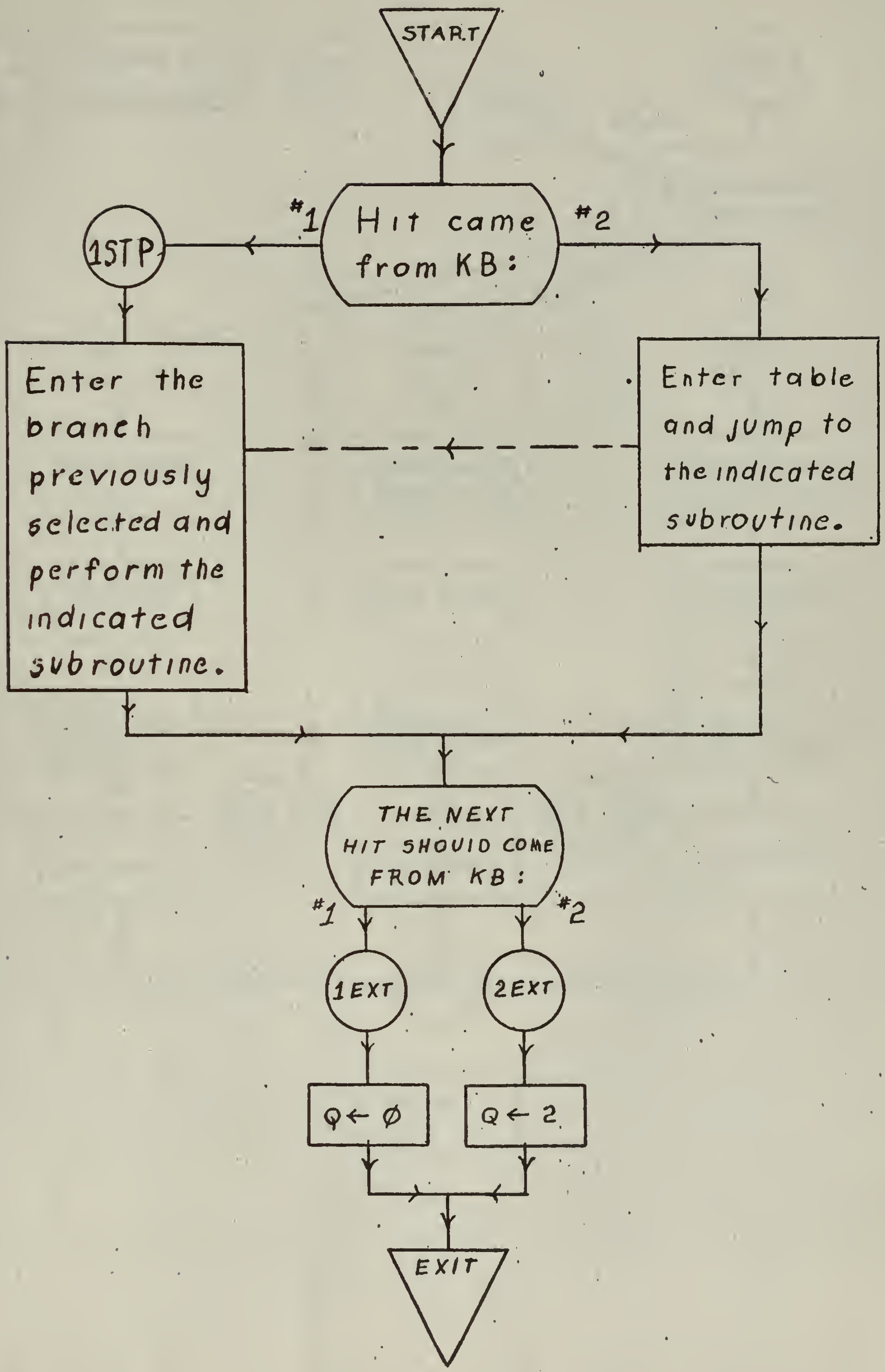
# DD65 Keyboards



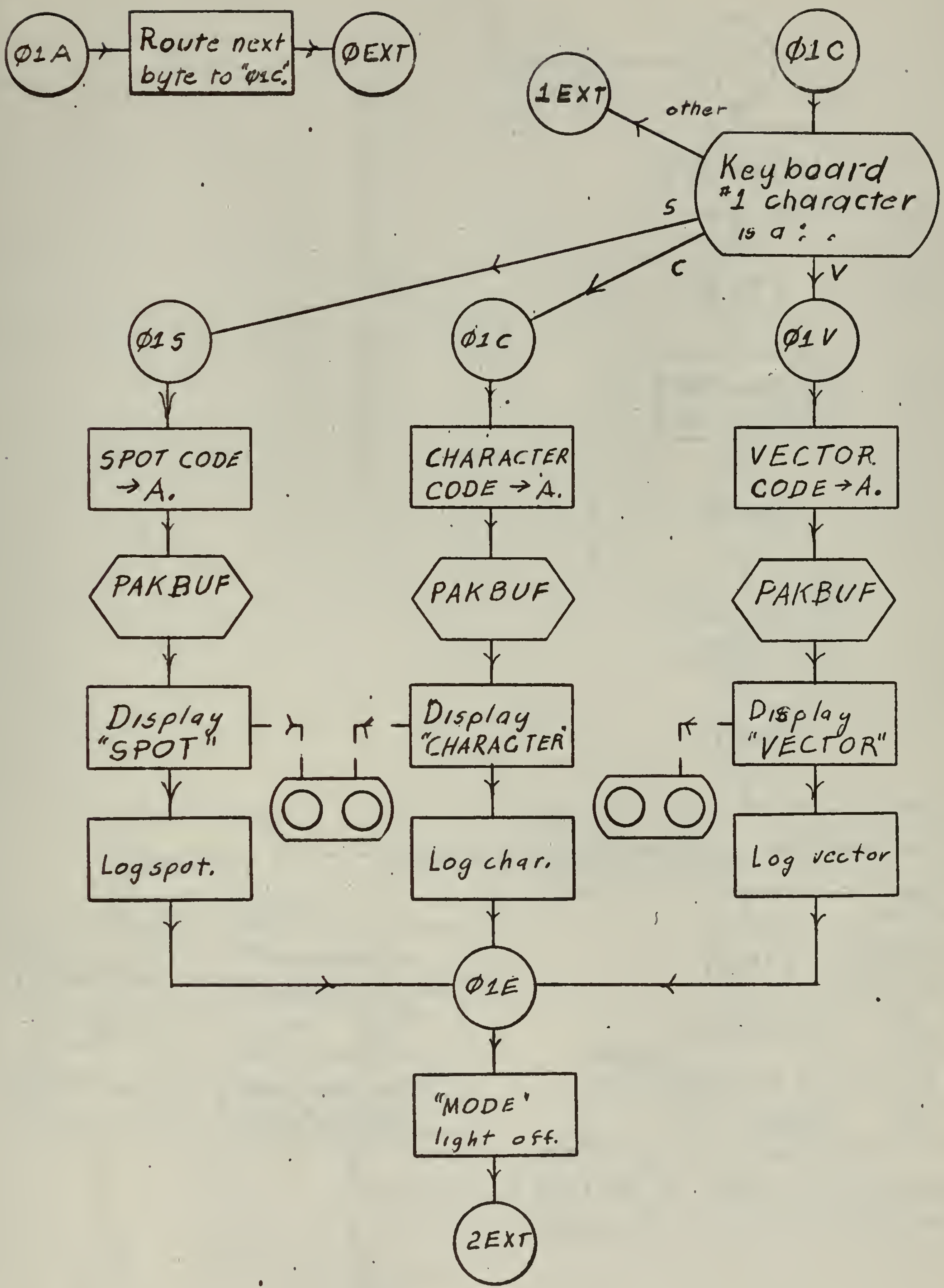
## DD65A


Start
Clear KB Hits and Int. Selects.
Clear PICBUF
Clear DD65
1ST
Was KB #2 Hit?
N
Y
Q : 0
≠
=
Q←2
RESET
A←Hit
KYBORD
2ST
Was KB #1 Hit?
N
Y
Q : 0
=
≠
A←Hit
KYBORD

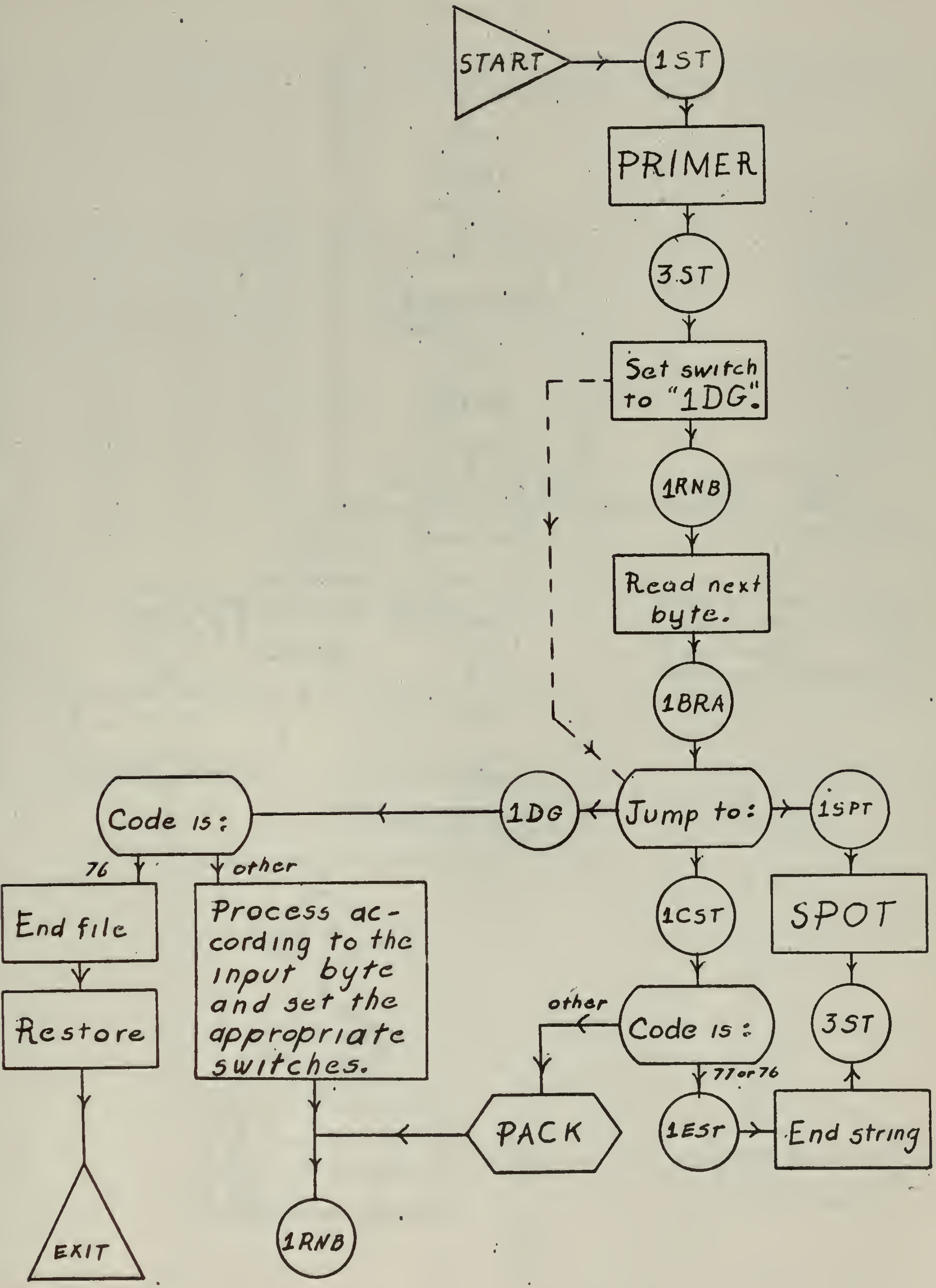
# KYBORD


START
Hit came from KB:
#1
#2
1STP
Enter the branch previously selected and perform the indicated subroutine.
Enter table and jump to the indicated subroutine.
THE NEXT HIT SHOUID COME FROM KB:
#1
#2
1EXT
2EXT
Q ← 0
Q ← 2
EXIT

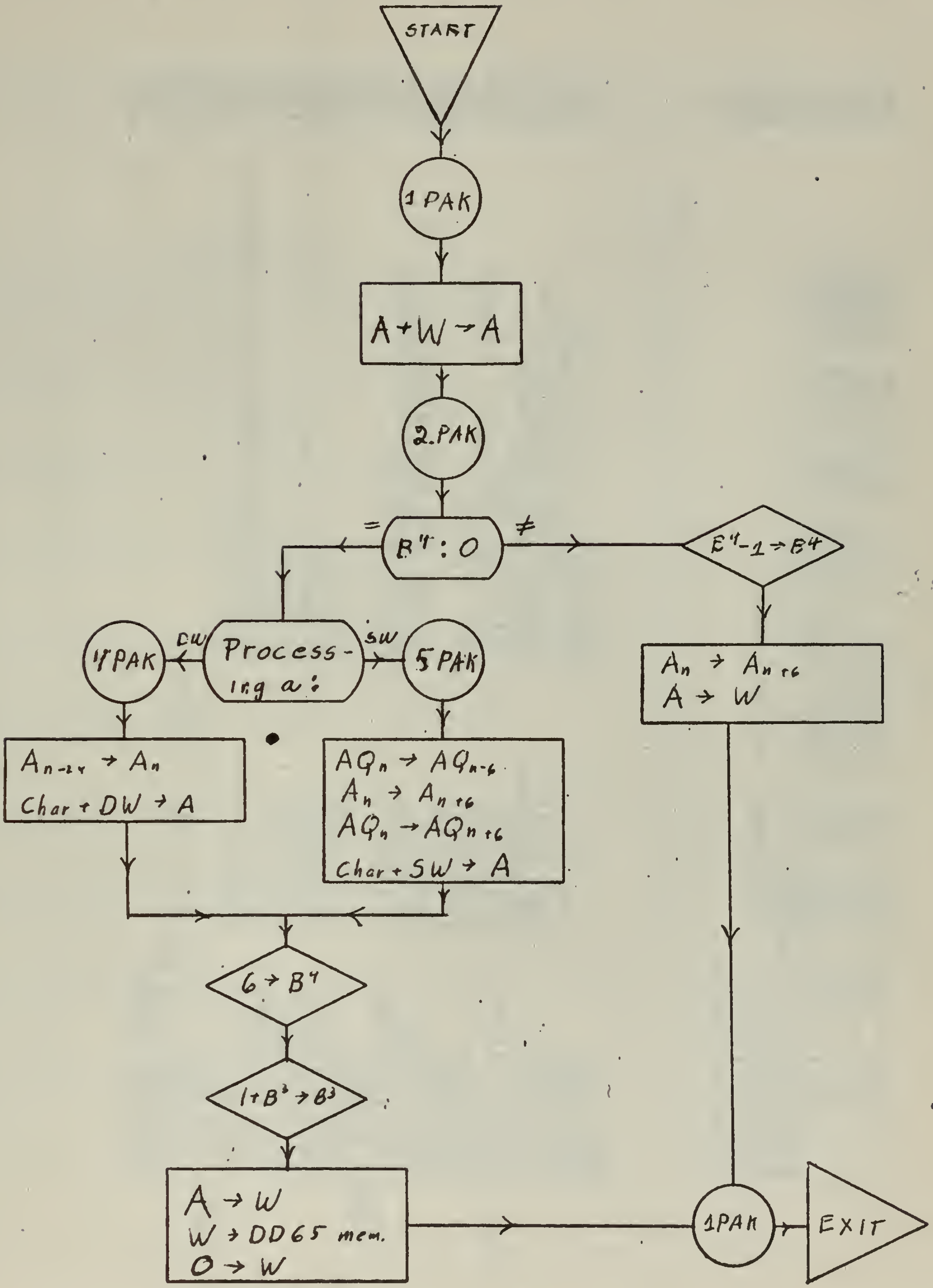
# MODE


Ø1A
Route next byte to "φ1C".
ØEXT
Ø1C
1EXT
other
Keyboard #1 character is a:
S
C
V
Ø1S
Ø1C
Ø1V
SPOT CODE → A.
CHARACTER CODE → A.
VECTOR CODE → A.
PAKBUF
PAKBUF
PAKBUF
Display "SPOT"
Display "CHARACTER"
Display "VECTOR"
Log spot.
Log char.
Log vector
Ø1E
"MODE" light off.
2EXT

# PICGEN


START
1ST
PRIMER
3ST
Set switch to "1DG".
1RNB
Read next byte.
1BRA
Jump to:
1DG
1SPT
Code is:
76
other
End file
Restore
EXIT
Process according to the input byte and set the appropriate switches.
1RNB
1CST
Code is:
other
77 or 76
PACK
1EST
End string
SPOT
3ST

# PACK


START
1 PAK
A+W→A
2.PAK
=
B⁴ : 0
≠
B⁴-1→B⁴
DW
Processing a:
SW
4PAK
5 PAK
An → An+6
A → W
An-24 → An
Char + DW → A
AQn → AQn-6
An → An+6
AQn → AQn+6
Char + SW → A
6 → B⁴
1+B³ → B³
A → W
W → DD65 mem.
0 → W
1PAK
EXIT

```
       MACHINE DD65A                                                      DD6
       LIB(KYBORD=KYB)                                                    DD6
       CON(MASK = 7777 7777 7777 7700B)                                   DD6
* PREP KEYBORD.                                                           DD6
 1QT   SLJ(N)          ENQ(2)          . LOAD FLAG.                       DD6
       EXF(77121B)     EXF(77111B)     . RELEASE AND REMOVE INT. SELECTS. DD6
       EXF7(77172B)    SLJ(L+2)        . IF KB1 HIT, EXIT.                DD6
       EXF(77140B)     INT(INWD)       . SEL KB1.                         DD6
       EXF7(77175B)    SLJ(L+2)        . IF KB2 HIT, EXIT.                DD6
       EXF(77120B)     INT(INWD)       . SEL KB2.                         DD6
       ENA(34B)        SLJ4(KYB)       . CLEAR.                           DD6
* INPUT CYCLE.                                                            DD6
 1ST   EXF7(77175B)    SLJ(2ST)        . KB2 HIT, EXIT.                   DD6
       EXF(77120B)     INT(INWD)       . SELECT AND INPUT FROM KB2.       DD6
       QJP1(L+2)       ENQ(2)          . IF KB2 NOT UP, RESET.            DD6
       ENA(55B)        SLJ4(KYB)       .                                  DD6
       LDA(INWD)       SCL(MASK)       . LOAD INPUT.                      DD6
       ENQ(2)          SLJ4(KYB)       . SET FLAG AND ENTER KYBORD.       DD6
 2ST   EXF7(77172B)    SLJ(1ST)        . KB1 HIT, EXIT.                   DD6
                       EXF(77140B)     . SELECT KB1 FOR INPUT.            DD6
       INT(INWD)       QJP1(1ST)       . KB1 HIT NOT PROPER, JUMP.        DD6
       LDA(INWD)       SLJ4(KYB)       . LOAD AND ENTER KYBORD.           DD6
       SLJ(1ST)                        . RECYCLE.                         DD6
       END


       MACHINE KYBORD                                                     PRK
       LOC( RST=5000)                                                     PRK
       LIB(CHALOC=C1, PICGEN=PF, FLASH=FLASH)                             PRK
       RSV(PICBUF = 1000B).                                               PRK
       CON(CHA1 =4557 6370 1670 1220B,CHA2 =6151 6163 2365 0051B,         PRK
1          VCT1 =4557 2565 1670 1220B,VCT2 =6323 4651 2020 0020B,         PRK
2          SPT1 =4557 2247 1670 1220B,SPT2 =4623 2020 2020 0020B,         PRK
3          SML1 =4557 2244 1674 1160B,SML2 =6143 4320 0102 0006B,         PRK
4          MED1 =4557 4465 1674 1160B,MED2 =6471 2444 2006 0003B,         PRK
```

```
5      LRG1 =4557 4361 1674 1160B,LRG2 =5167 6520 2003 0001B,          PRK
6      RHT1 =4557 5123 1700 1120B,RHT2 =7320 2324 6265 0020B,          PRK
7      LFT1 =4557 4365 1700 1120B,LFT2 =6623 2023 2462 0065B,          PRK
8      NOR1 =4557 4546 1704 1060B,NOR2 =5144 6143 2020 0020B)          PRK
 CON(BRT1 =4557 6251 1704 1060B,BRT2 =7167 7023 2020 0020B,          PRK
1      CPL3 =2000 0000 2020 0020B,CPL2 =6151 2143 7145 0065B,          PRK
2      CPL1 =4557 6370 1662 1757B,DWN2 =6351 2064 4626 0045B,          PRK
3      RIT1 =4557 7145 1714 1717B,RIT2 =6351 2035 2020 0020B,          PRK
4      SG1  =4557 2267 1710 1020B,SG2  =7320 2247 6163 0065B,          PRK
5      DB1  =4557 6462 1710 1020B,TP1  =4557 2347 1710 1020B,          PRK
6      RM2  =5144 2000 0000 4020B,RM1  =0000 5520 5730 5374B,          PRK
7      LM2  =4344 2000 0000 4020B,LM1  =0000 5520 5734 5374B)          PRK
 CON(TB1 = 2000 5720 1740 5376B, TB2 = 2000 5720 1742 5376B,         PRK
1      TB3 = 2000 5720 1744 5376B, TB4 = 2000 5720 1746 5376B,         PRK
2      TB5 = 2000 5720 1750 5376B, TB6 = 2000 5720 1752 5376B,         PRK
3      TB7 = 2000 5720 1754 5376B, TB8 = 2000 5720 1756 5376B,         PRK
4      TB9 = 2000 5720 1760 5376B, TB10= 2000 5720 1762 5376B,         PRK
5      TB11= 2000 5720 1764 5376B, TB12= 2000 5720 1766 5376B,         PRK
6      TB13= 2000 5720 1770 5376B, TB14= 2000 5720 1772 5376B,         PRK
7      TB15= 2000 5720 1774 5376B, TB16= 2000 5720 1776 5376B,         PRK
8      PIP  =2000 2720 1720 5000B,MARKER=5000 0000 1722 5000B)         PRK
 CON(ERR2 =0120 6551 5145 4051B,ERR1A=0755 4262 1724 5577B,          PRK
1      ERR1B=2733 4262 1724 5577B,ERR1C=5667 4262 1724 5577B,          PRK
2      ERC2 =0,ERC1 =17241000B,MK11 =4000B,                             PRK
3      TERM =7777 7777 7776 7776B,FUL3 =2066 2443 4373 4020B,          PRK
4      FUL2 =6362 2466 2071 4022B,FUL1 =5617 4771 5656 5000B,          PRK
5      MKXY =7000 7777 7777 7000B,MKX  =7000 7777 7777 7777B,          PRK
6      MKA  =7700 0000 7777 7777B,MKN  =7777 7700 0000 7777B,          PRK
7      SEV  =7777 7777 7777 7777B,SPAC = 1,                             PRK
8      BLANK=2020 2020 2020 2020B,M6   =6B)                             PRK
 CON(NPOSIT = 60 0000 0001B,      PIPOFF = 1720 5000B,               PRK
1      TL1 = 377B,  TL2 = 377B,    TL3 = 377B,  TL4  = 377B,           PRK
2      TL5 = 377B,  TL6 = 377B,    TL7 = 377B,  TL8  = 377B,           PRK
3      TL9 = 377B,  TL10= 377B,    TL11= 377B,  TL12= 377B,            PRK
4      TL13= 377B,  TL14= 377B,    TL15= 377B,  TL16= 377B,            PRK
5      TL17= 377B,                                                      PRK
6      SIZE=4B,VSIZE=4B,MKCLR=76B,MK4=10B,MK5=20B,MK6=40B,MK77=77B,    PRK
7      TDW= 5720 0000 5377B,MKT=7000 7777 0000 7777B,MKME=4 0000B)     PRK
 CON(K0=77B,K1=7700B,K2=77 0000B,K3=7700 0000B,K4=77 0000 0000B,     PRK
1      K5=7700 0000 0000B,K6=77 0000 0000 0000B,                        PRK
2      K7=7700 0000 0000 0000B)                                         PRK
*  DECODER.                                                             MAK
 1ST  SLJ(N)          QJP1(L+2)      . KB2 INPUT, JUMP.                 MAK
 1STP SLJ(N)                         . KB1 BRANCH POINT.                MAK
    EXF(77741B)       INA(-1)        . RESET OFF.                       MAK
    LRS(3)            SAU(L+1)       . SHORTEN TABLE BY EXTRACTING      MAK
    ENI1(N)           LLS(3)         .      DONT CARES.                 MAK
    IJP1(L+1)         SLJ(L+2)       .                                  MAK
    INA(-3)           SLJ(L-1)       .                                  MAK
```

```
         INA(1TBL)       SAU(L+1)        .                               MAK
         SLJ(N)                          . ENTER TABLE.                  MAK
1TBL     EXF(77202B)     SLJ(01A)        . LIGHT MODE.                   MAK
         EXF(77204B)     SLJ(02A)        . SIZE.                         MAK
         EXF(77210B)     SLJ(03A)        . TUBE.                         MAK
         EXF(77220B)     SLJ(04A)        . INTENSITY.                    MAK
         EXF(77240B)     SLJ(05A)        . INCREMENT.                    MAK
         EXF(77302B)     SLJ(11A)        . LEFT MARGIN.                  MAK
         EXF(77304B)     SLJ(12A)        . RIGHT MARGIN.                 MAK
         EXF(77310B)     SLJ(13A)        . TAB SET.                      MAK
         EXF(77320B)     SLJ(14A)        . TAB RESET.                    MAK
         EXF(77340B)     SLJ(15A)        . SPACING.                      MAK
         EXF(77402B)     SLJ(2EXT)       .                               MAK
         EXF(77404B)     SLJ(2EXT)       .                               MAK
         EXF(77410B)     SLJ(2EXT)       .                               MAK
         EXF(77420B)     SLJ(2EXT)       .                               MAK
         EXF(77440B)     SLJ(2EXT)       .                               MAK
         EXF(77502B)     SLJ(31A)        . TYPE.                         MAK
         EXF(77504B)     SLJ(32A)        . START.                        MAK
         EXF(77510B)     SLJ(33A)        . MARKER.                       MAK
         EXF(77520B)     SLJ(34A)        . CLEAR.                        MAK
         EXF(77540B)     SLJ(35A)        . TEST LIGHTS.                  MAK
         EXF(77602B)     SLJ(41A)        . PIP.                          MAK
         SLJ(42A)                        . TRACK BALL.                   MAK
         EXF(77610B)     SLJ(2EXT)       .                               MAK
         EXF(77620B)     SLJ(44A)        . CHARACTER EDIT.               MAK
         EXF(77640B)     SLJ(2EXT)       .                               MAK
         EXF(77702B)     SLJ4(RST)       . RESTART MACHINE DD65A.        MAK
         EXF(77704B)     SLJ(2EXT)       .                               MAK
         EXF(77710B)     SLJ(2EXT)       .                               MAK
         EXF(77720B)     SLJ(54A)        . LINE EDIT.                    MAK
         EXF(77740B)     SLJ(55A)        . RESET.                        MAK
         SLJ(61A)                        . UP.                           MAK
         SLJ(62A)                        . RIGHT.                        MAK
         SLJ(63A)                        . DOWN.                         MAK
         SLJ(64A)                        . LEFT.                         MAK
* EXITS.                                                                 MAK
0EXT     ENQ(0)          SLJ(1ST)        . SET FLAG10 = 0 (KB1).         MAK
1EXT     ENI2(2)                         . KEYBOARD 1 ERROR EXIT.        MAK
         EXF(77010B)     EXF7(77010B)    . SEL AND WAIT FOR DD65 MEM.    MAK
         ENI3(2)         OUT3(ERR2)      .                               MAK
                         SLJ4(3WAT)      .                               MAK
         EXF(77010B)     EXF7(77010B)    . SEL AND WAIT FOR DD65 MEM.    MAK
         OUT(ERR1B)      SLJ4(3WAT)      .                               MAK
         EXF(77010B)     EXF7(77010B)    . SEL AND WAIT FOR DD65 MEM.    MAK
         OUT(ERR1C)      SLJ4(3WAT)      .                               MAK
         EXF(77010B)     EXF7(77010B)    . SEL AND WAIT FOR DD65 MEM.    MAK
         ENI3(2)         OUT3(ERC2)      .                               MAK
                         SLJ4(3WAT)      .                               MAK
```

```
      IJP2(1EXT+1)      SLJ(0EXT)          .                                          MAK
2EXT  ENQ(2)            SLJ(1ST)           . SET FLAG10 = 2 (KB2).                    MAK
* MODE - C=CHARACTER , V= VECTOR, S=SPOT.                                              01K
01A   ENA(01C)                             . ROUTE NEXT HIT.                          01K
01AE  SAU(1STP)         SLJ(0EXT)          .                                          01K
01C   ENI6(1)           INA(-63B)          . NO PICGEN.                               01K
      AJP(01CH)         INA(36B)           . CHARACTER.                               01K
      AJP(01V)          INA(3)             . VECTOR.                                  01K
      AJP(01S)          SLJ(1EXT)          . SPOT.                                    01K
01CH  ENA(10B)          SLJ4(1PB)          .                                          01K
      EXF(77010B)       EXF7(77010B)       . SEL AND WAIT FOR DD65 MEM.               01K
                        OUT(CHA1)          .                                          01K
      EXF(77010B)       EXF7(77010B)       . SEL AND WAIT FOR DD65 MEM.               01K
      OUT(CHA2)         ENA(0)             . MODE = 0-CA, 1-VCT, 2-SPOT.              01K
01E   STA(MODE)         EXF(77203B)        . EXIT MODE.                               01K
      SLJ(2EXT)                            .                                          01K
01V   ENA(11B)          SLJ4(1PB)          .                                          01K
      EXF(77010B)       EXF7(77010B)       . SEL AND WAIT FOR DD65 MEM.               01K
                        OUT(VCT1)          .                                          01K
      EXF(77010B)       EXF7(77010B)       . SEL AND WAIT FOR DD65 MEM.               01K
                        OUT(VCT2)          .                                          01K
      ENA(1)            SLJ(01E)           .                                          01K
01S   ENA(12B)          SLJ4(1PB)          .                                          01K
      EXF(77010B)       EXF7(77010B)       . SEL AND WAIT FOR DD65 MEM.               01K
                        OUT(SPT1)          .                                          01K
      EXF(77010B)       EXF7(77010B)       . SEL AND WAIT FOR DD65 MEM.               01K
                        OUT(SPT2)          .                                          01K
      ENA(2)            SLJ(01E)           .                                          01K
* SIZE - S=SMALL, M=MEDIUM, L=LARGE.                                                   02K
02A   ENA(02C)          SLJ(01AE)          . ROUTE NEXT HIT.                          02K
02C   ENI6(1)           INA(-22B)          .                                          02K
      AJP(02S)          INA(-22B)          . SMALL.                                   02K
      AJP(02M)          INA(1)             . MEDIUM.                                  02K
      AJP(02L)          SLJ(1EXT)          . LARGE.                                   02K
02S   ENA(30B)          SLJ4(1PB)          . SIZE = 4-SMALL, 8-MED, 16-LARGE.         02K
      EXF(77010B)       EXF7(77010B)       . SEL AND WAIT FOR DD65 MEM.               02K
                        OUT(SML1)          .                                          02K
      EXF(77010B)       EXF7(77010B)       . SEL AND WAIT FOR DD65 MEM.               02K
      OUT(SML2)         ENA(4)             .                                          02K
02E   STA(SIZE)         ARS(1)             . CAL SIZE / 2 = SIZE2.                    02K
      STA(SIZE2)        SLJ4(15Z)          . RECAL SPACING.                           02K
                        SLJ4(1MAR)         . RECAL CHAR/LINE.                         02K
      EXF(77205B)       SLJ(2EXT)          . EXIT.                                    02K
02M   ENA(31B)          SLJ4(1PB)          .                                          02K
      EXF(77010B)       EXF7(77010B)       . SEL AND WAIT FOR DD65 MEM.               02K
                        OUT(MED1)          .                                          02K
      EXF(77010B)       EXF7(77010B)       . SEL AND WAIT FOR DD65 MEM.               02K
                        OUT(MED2)          .                                          02K
      ENA(10B)          SLJ(02E)           .                                          02K
```

```
 02L   ENA(32B)        SLJ4(1PB)       .                                       02K
       EXF(77010B)     EXF7(77010B)    .  SEL AND WAIT FOR DD65 MEM.           02K
                       OUT(LRG1)       .                                       02K
       EXF(77010B)     EXF7(77010B)    .  SEL AND WAIT FOR DD65 MEM.           02K
                       OUT(LRG2)       .                                       02K
       ENA(20B)        SLJ(02E)        .                                       02K
* TUBE - R=RIGHT, L=LEFT.                                                      03K
 03A   ENA(03C)        SLJ(01AE)       .  ROUTE NEXT HIT.                      03K
 03C   ENI6(1)         INA(-51B)       .  RIGHT.                               03K
       AJP(03R)        INA(6)          .  LEFT.                                03K
       AJP(03L)        SLJ(1EXT)       .  NOTA.                                03K
 03R   ENA(40B)        SLJ4(1PB)       .                                       03K
       ENI1(35B)                       .  PLACE INDICATORS ON LEFT TUBE.       03K
       LDA1(CHA1)      SCL(MK11)       .                                       03K
       STA1(CHA1)      IJP1(L-1)       .                                       03K
       ENI1(27B)                       .                                       03K
       LDA1(RM2)       SST(MK11)       .                                       03K
       STA1(RM2)       IJP1(L-1)       .                                       03K
       EXF(77010B)     EXF7(77010B)    .  SEL AND WAIT FOR DD65 MEM.           03K
                       OUT(RHT1)       .                                       03K
       EXF(77010B)     EXF7(77010B)    .  SEL AND WAIT FOR DD65 MEM.           03K
                       OUT(RHT2)       .                                       03K
       EXF(77211B)     SLJ(2EXT)       .  LIGHT OFF AND EXIT.                  03K
 03L   ENA(41B)        SLJ4(1PB)       .                                       03K
       ENI1(35B)                       .  PLACE INDICATORS ON RIGHT TUBE.      03K
       LDA1(CHA1)      SST(MK11)       .                                       03K
       STA1(CHA1)      IJP1(L-1)       .                                       03K
       ENI1(27B)                       .                                       03K
       LDA1(RM2)       SCL(MK11)       .                                       03K
       STA1(RM2)       IJP1(L-1)       .                                       03K
       EXF(77010B)     EXF7(77010B)    .  SEL AND WAIT FOR DD65 MEM.           03K
                       OUT(LFT1)       .  OUTPUT.                              03K
       EXF(77010B)     EXF7(77010B)    .  SEL AND WAIT FOR DD65 MEM.           03K
                       OUT(LFT2)       .                                       03K
       EXF(77211B)     SLJ(2EXT)       .  LIGHT OFF AND EXIT.                  03K
* INTENSITY - N=NORMAL, B=BRIGHT.                                              04K
 04A   ENA(04C)        SLJ(01AE)       .  ROUTE NEXT HIT.                      04K
 04C   ENI6(1)         INA(-45B)       .  NORMAL.                              04K
       AJP(04N)        INA(-15B)       .  BRIGHT.                              04K
       AJP(04G)        SLJ(1EXT)       .  NOTA.                                04K
 04N   ENA(50B)        SLJ4(1PB)       .                                       04K
       EXF(77010B)     EXF7(77010B)    .  SEL AND WAIT FOR DD65 MEM.           04K
                       OUT(NOR1)       .                                       04K
       EXF(77010B)     EXF7(77010B)    .  SEL AND WAIT FOR DD65 MEM.           04K
                       OUT(NOR2)       .                                       04K
       EXF(77221B)     SLJ(2EXT)       .  INTENSITY OFF.                       04K
 04G   ENA(51B)        SLJ4(1PB)       .                                       04K
       EXF(77010B)     EXF7(77010B)    .  SEL AND WAIT FOR DD65 MEM.           04K
                       OUT(BRT1)       .                                       04K
```

```
        EXF(77010B)      EXF7(77010B)    . SEL AND WAIT FOR DD65 MEM.           04K
                         OUT(BRT2)       .                                      04K
        EXF(77221B)      SLJ(2EXT)       . INTENSITY OFF.                       04K
* INCREMENT - R=RIGHT, D=DOWN.                                                  05K
 05A    ENA(05C)         SLJ(01AE)       . ROUTE NEXT HIT.                      05K
 05C    ENI6(1)          INA(-51B)       . RIGHT.                               05K
        AJP(05R)         INA(-13B)       . DOWN.                                05K
        AJP(05D)         SLJ(1EXT)       . NOTA.                                05K
 05R    ENA(60B)         SLJ4(1PB)       .                                      05K
        EXF(77010B)      EXF7(77010B)    . SEL AND WAIT FOR DD65 MEM.           05K
                         OUT(RIT1)       .                                      05K
        EXF(77010B)      EXF7(77010B)    . SEL AND WAIT FOR DD65 MEM.           05K
        OUT(RIT2)        ENA(-1)         . INCR = -1 = RIGHT.                   05K
 05E    STA(INCR)        EXF(77241B)     . EXIT.                                05K
        SLJ(2EXT)                        .                                      05K
 05D    ENA(61B)         SLJ4(1PB)       .                                      05K
        EXF(77010B)      EXF7(77010B)    . SEL AND WAIT FOR DD65 MEM.           05K
                         OUT(RIT1)       .                                      05K
        EXF(77010B)      EXF7(77010B)    . SEL AND WAIT FOR DD65 MEM.           05K
                         OUT(DWN2)       .                                      05K
        ENA(1)           SLJ(05E)        . INCR = 1 = DOWN.                     05K
* LEFT  MARGIN = P-USE PIPX FOR INPUT, + OR - AND 2 OCT DIGITS.                 11K
 11A    ENA(11C)         SLJ(01AE)       .                                      11K
 11C    INA(-47B)        AJP(11PI)       . PIP.                                 11K
        INA(7)           AJP1(L+2)       . NOT - SIGN, JUMP.                    11K
        ENA(11N)         SLJ(01AE)       . -.                                   11K
        INA(-20B)        AJP1(1EXT)      . NOT + SIGN, ERROR EXIT.              11K
        ENA(11P)         SLJ(01AE)       . +.                                   11K
 11PI   LDA(PIPX)        ARS(2)          . PIPPER.                              11K
        ALS(2)           SLJ(11E)        . USE ONLY POSITS / BY 4.              11K
 11N                     SLJ4(1RET)      . NEG. INPUT.                          11K
        SCM(SEV)         SLJ(11E)        .                                      11K
 11P                     SLJ4(1RET)      . POS. INPUT.                          11K
 11E    STA(LMAR)        SLJ4(1MAR)      .                                      11K
        EXF(77303B)      SLJ(2EXT)       . EXIT.                                11K
* RIGHT MARGIN (SEE LEFT MARGIN).                                               12K
 12A    ENA(12C)         SLJ(01AE)       .                                      12K
 12C    INA(-47B)        AJP(12PI)       .                                      12K
        INA(7)           AJP1(L+2)       .                                      12K
        ENA(12N)         SLJ(01AE)       .                                      12K
        INA(-20B)        AJP1(1EXT)      .                                      12K
        ENA(12P)         SLJ(01AE)       .                                      12K
 12PI   LDA(PIPX)        ARS(2)          . PIP.                                 12K
        ALS(2)           SLJ(12E)        . USE ONLY POSITS / BY 4.              12K
 12N                     SLJ4(1RET)      .                                      12K
        SCM(SEV)         SLJ(12E)        .                                      12K
 12P                     SLJ4(1RET)      .                                      12K
 12E    STA(RMAR)        SLJ4(1MAR)      .                                      12K
        EXF(77305B)      SLJ(2EXT)       . EXIT.                                12K
```

```
* TAB SET - PIPPER IS USED (PIPX).                                            13K
 13A  LDA(PIPX)      ARS(2)         . NEXT TAB IS PIPER X LOCATION.           13K
      ALS(2)         STA(T)         . ONLY POSITS / BY 4 ARE USED.            13K
      ENI1(0)                       . ZERO INDIX.                             13K
      LDA1(TL1)      SUB(T)         . OLD TAB, EXIT.                          13K
      AJP(1PTL)      AJP3(L+3)      . NEW TAB TOO FAR TO RIGHT, JUMP.         13K
      LDA(T)         LDQ1(TL1)      . INSERT NEW TAB.                         13K
      STA1(TL1)      STQ(T)         .                                         13K
      ISK1(17B)      SLJ(L-4)       .                                         13K
      LDA(PIPX)      STA(TL16)      . IN ANY EVENT STORE IN TL16.             13K
      SLJ(1PTL)                     .                                         13K
* TAB CLEAR - PIPPER IS USED.                                                 14K
 14A  ENI1(0)                       .                                         14K
      LDA(PIPX)      SUB1(TL1)      .                                         14K
      AJP3(L+2)      LDA(PIPX)      . TRY NEXT TAB IF NEG.                    14K
      SUB1(TL2)      AJP3(14CL)     . DELETE IF ONLY ONE.                     14K
      ISK1(17B)      SLJ(L-3)       .                                         14K
      SLJ(1PTL)                     . NONE FOUND.                             14K
 14CL LDA1(TL2)      STA1(TL1)      . EXTRACT EXTRA TAB.                      14K
      ISK1(17B)      SLJ(L-1)       .                                         14K
      SLJ(1PTL)                     .                                         14K
* SPACING - S=SINGLE, D=DOUBLE, T= TRIPLE.                                    15K
 15A  ENA(15C)       SLJ(01AE)      .                                         15K
 15C  INA(-22B)      AJP(15S)       . SINGLE.                                 15K
      INA(-42B)      AJP(15D)       . DOUBLE.                                 15K
      INA(41B)       AJP(15T)       . TRIPLE                                  15K
      SLJ(1EXT)                     . NOTA.                                   15K
 15S  ENA(1)         STA(SPAC)      . DELTA = SPACING * SIZE (528).           15K
      EXF(77010B)    EXF7(77010B)   . SEL AND WAIT FOR DD65 MEM.              15K
      OUT(SG1)       SLJ(15E)       .                                         15K
 15D  ENA(2)         STA(SPAC)      .                                         15K
      EXF(77010B)    EXF7(77010B)   . SEL AND WAIT FOR DD65 MEM.              15K
      OUT(DB1)       SLJ(15E)       .                                         15K
 15T  ENA(3)         STA(SPAC)      .                                         15K
      EXF(77010B)    EXF7(77010B)   . SEL AND WAIT FOR DD65 MEM.              15K
      OUT(TP1)       SLJ(15E)       .                                         15K
 15E  EXF(77010B)    EXF7(77010B)   . SEL AND WAIT FOR DD65 MEM.              15K
      OUT(SG2)       SLJ4(15Z)      . RECAL SPACING.                          15K
      EXF(77341B)    SLJ(2EXT)      . EXIT.                                   15K
* TYPE - KB1 ACTIVATED IN CURRENT MODE.                                       31K
 31A  LDA(MODE)      AJP(L+3)       . CHAR.                                   31K
      INA(-2)        AJP3(31V)      . VECTOR.                                 31K
      AJP(31SA)      SLJ(1EXT)      . SPOT.                                   31K
      ENA(31CA)                     . ROUTE CHAR.                             31K
 31C1 SAU(1STP)      SLJ4(1LOC)     . POSITION.                               31K
      ENA(0)         SLJ4(1PB)      . START STRING.                           31K
                     SLJ4(33MK)     . MARKER                                  31K
      SLJ(0EXT)                     .                                         31K
 31V  ENA(31VA)      SAU(1STP)      . ROUTE VECTOR.                           31K
```

```
      SLJ(31C1)                     .                                   31K
* MAIN CHARACTER LOOP.                                                  31K
 31CA AJP(1BKS)      INA(-36B)      . BACKSPACE, JUMP.                  31K
      AJP(31CT)      INA(-40B)      . TAB, JUMP.                        31K
      AJP(31CR)      INA(-1)        . CR, JUMP.                         31K
      AJP(0EXT)      INA(77B)       . SEMICLAN NOT ALLOWED, RESTORE CODE31K
      ENI6(0)        SLJ4(1PB)      . PRINT CHARACTER.                  31K
                     SLJ4(31CM)     . CAL MAR AND MARKER.               31K
                     SLJ4(33MK)     . MARKER.                           31K
      SLJ(0EXT)                     . EXIT.                             31K
 31CT ENI1(0)        ENI6(0)        . TAB.                              31K
      LDA(INCR)      AJP2(31CT1)    . TAB DOWN, JUMP.                   31K
      LDA1(TL1)      ADD(SIZE2)     .                                   31K
      SUB(XLOC)      AJP2(L+3)      . TAB TOO FAR TO LEFT, JUMP.        31K
      ISK1(17B)      SLJ(L-2)       .                                   31K
      SLJ(1EXT)                     . ERROR EXIT.                       31K
      SUB(SIZE)      AJP3(31CA+6)   . JUST RIGHT, EXIT.                 31K
      ENA(20B)       SLJ4(1PB)      . LOAD BLANK, TRY AGAIN.            31K
                     SLJ4(31CM)     . PROCESS.                          31K
      SLJ(31CT)                     .                                   31K
31CT1 LDA(YLOC)      SUB(SIZE2)     . TAB DOWN.                         31K
      SUB1(TL1)      AJP3(L+3)      .                                   31K
      ISK1(17B)      SLJ(L-2)       .                                   31K
      SLJ(1EXT)                     . ERROR EXIT.                       31K
      IJP(31CT+7)    SLJ(31CR)      .                                   31K
 31CM SLJ(N)         LDA(SIZE)      .                                   31K
      LDQ(INCR)      QJP2(31CM1)    . INCR DOWN, JUMP.                  31K
      RAD(XLOC)      SUB(RMAR)      .                                   31K
      AJP3(31CM)     LDA(DELTA)     . INSIDE MAR, JUMP.                 31K
      RSB(YLOC)      LDA(LMAR)      . START NEW LINE)                   31K
      STA(XLOC)      ENI6(2)        .                                   31K
      ENA(77B)       SLJ4(1PB)      .                                   31K
                     SLJ4(1LOC)     .                                   31K
      ENI6(0)        SLJ4(33MK)     .                                   31K
      ENA(0)         SLJ4(1PB)      .                                   31K
      SLJ(0EXT)                     . CR EXIT.                          31K
31CM1 RSB(YLOC)      INA(377B)      .                                   31K
      AJP2(31CM)     ENA(376B)      . INSIDE MAR, JUMP.                 31K
      STA(YLOC)      LDA(DELTA)     .                                   31K
      ADD(XLOC)      SLJ(31CM+5)    .                                   31K
 31CR LAC(INCR)                     . CR.                               31K
      AJP3(31CM1+1)  SLJ(31CM+3)    .                                   31K
* VECTOR - S=SMALL, M=MEDIUM, L=LARGE, E=EXTRA LARGE, 0=BLANK, 5=UNBLANK31K
*     9=NE, 6=E, 3=SE, 2=S, 1=SW, 4=W, 7=NW, 8=NORTH.                   31K
 31VA AJP(1EXT)      INA(-5)        . 0, EXIT.                          31K
      AJP(31VU)      INA(-5)        . UNBLANK, JUMP.                    31K
      AJP(31VB)      INA(-10B)      . BLANK.                            31K
      AJP(31VS)      INA(-21B)      . SMALL.                            31K
      AJP(31VL)      INA(-1)        . LARGE.                            31K
```

```
     AJP(31VM)   INA(-21B)     . MEDIUM.                           31K
     AJP(31VX)   INA(53B)      . EXTRA LARGE.                      31K
     AJP2(1EXT)  INA(12B)      . NOTA AND GT 12, EXIT.   RESTORE A.31K
     INA(L+1)    SAU(L+1)      . PREP TABLE.                       31K
     SLJ(N)                    .                                   31K
     ENA(5)      SLJ(31VP)     . SW.                               31K
     ENA(6)      SLJ(31VP)     . S.                                31K
     ENA(7)      SLJ(31VP)     . SE.                               31K
     ENA(4)      SLJ(31VP)     . W.                                31K
31VT ZRO(0)      ZRO(0)        . BLANK AND SIZE STORAGE.           31K
     ENA(0)      SLJ(31VP)     . E.                                31K
     ENA(3)      SLJ(31VP)     . NW.                               31K
     ENA(2)      SLJ(31VP)     . N.                                31K
     ENA(1)      SLJ(31VP)     . NE                                31K
* PRINT VECTOR.                                                    31K
31VP STA(T)      ADD(31VT)     . HOLD DIRECTION. ADD INT AND SIZE. 31K
     STA(TEMP)   LDA(VSIZE)    . TEMP = OUTPUT CELL.               31K
     INA(-20B)   AJP(L+4)      . IF LARGE, JUMP.                   31K
     AJP2(L+7)   ENI6(0)       . IF EXTRA LARGE, JUMP.             31K
     LDA(TEMP)   SLJ4(1PB)     . PRINT SMALL AND MEDIUM VECTORS.   31K
     SLJ(31V9)                 . TRACK SMALL AND MEDIUM VECTORS.   31K
     ENI6(1)                   . LARGE. PRINT 1 OF 2 MEDIUMS.      31K
     LDA(TEMP)   SLJ4(1PB)     .                                   31K
     LDA(TEMP)   SLJ4(1PB)     .                                   31K
     SLJ(31V9)                 . TRACK LARGE.                      31K
     ENI6(3)                   . EXTRA LARGE. PRINT 1 OF 3 MEDIUMS.31K
     LDA(TEMP)   SLJ4(1PB)     .                                   31K
     LDA(TEMP)   SLJ4(1PB)     .                                   31K
     LDA(TEMP)   SLJ4(1PB)     .                                   31K
     LDA(TEMP)   SLJ4(1PB)     .                                   31K
     SLJ(31V9)                 . TRACK EXTRA LARGE.                31K
* TRACK  VECTOR WITH MARKER.                                       31K
31V9 LDA(T)      INA(-5)       . ALL SOUTHS.                       31K
                 AJP6(31MY)    .                                   31K
     LDA(T)      AJP(L+2)      . ALL NORTHS.                       31K
     INA(-4)     AJP7(31PY)    .                                   31K
     LDA(T)      INA(-3)       . ALL WESTS.                        31K
     AJP3(L+2)   INA(-3)       .                                   31K
                 AJP7(31MK)    .                                   31K
     LDA(T)      AJP(L+3)      . ALL EASTS.                        31K
     INA(-1)     AJP(L+2)      .                                   31K
     INA(-6)     AJP1(L+2)     .                                   31K
                 SLJ4(31PX)    .                                   31K
31VE             SLJ4(33MK)    .                                   31K
     SLJ(0EXT)                 .                                   31K
31PX SLJ(N)      LDA(VSIZE)    . +X.                               31K
     RAD(XLOC)   SLJ(L-1)      .                                   31K
31MX SLJ(N)      LDA(VSIZE)    . -X.                               31K
     RSB(XLOC)   SLJ(L-1)      .                                   31K
```

```
 31PY SLJ(N)         LDA(VSIZE)     . +Y.                               31K
      RAD(YLOC)      SLJ(L-1)       .                                   31K
 31MY SLJ(N)         LDA(VSIZE)     . -Y.                               31K
      RSB(YLOC)      SLJ(L-1)       .                                   31K
* MODIFY VECTOR.                                                        31K
 31VU LDA(31VT)      SST(MK6)       . UNBLANK.                          31K
      STA(31VT)      SLJ(0EXT)      .                                   31K
 31VB LDA(31VT)      SCL(MK6)       . BLANK.                            31K
      STA(31VT)      SLJ(0EXT)      .                                   31K
 31VS LDA(31VT)      SCL(MK5)       . SMALL.                            31K
      SCL(MK4)       ENQ(4)         .                                   31K
      STQ(VSIZE)     SLJ(0EXT)      . EXIT.                             31K
 31VM ENQ(10B)                      . MEDIUM.                           31K
 31V1 STQ(VSIZE)     LDA(31VT)      .                                   31K
      SST(MK4)       SCL(MK5)       .                                   31K
      STA(31VT)      SLJ(0EXT)      . EXIT.                             31K
 31VL ENQ(20B)       SLJ(31V1)      . LARGE.                            31K
 31VX ENQ(40B)       SLJ(31V1)      . EXTRA LARGE.                      31K
* SPOT.                                                                 31K
 31SA ENI6(4)        SLJ4(1PB)      . PRINT WHEN SPOT COMPLETE, START   31K
      LDA(PIPX)      ARS(3)         .      STRING.                      31K
                     SLJ4(1PB)      . 1ST 6 BITS.                       31K
      LDA(PIPY)      LRS(11B)       .                                   31K
      LDA(PIPX)      LLS(3)         .                                   31K
                     SLJ4(1PB)      . 2ND 6 BITS.                       31K
      LDA(PIPY)      SLJ4(1PB)      . 3RD 6 BITS.                       31K
      ENI6(0)                       . PRINT.                            31K
      ENA(77B)       SLJ4(1PB)      . END STRING.                       31K
      EXF(77503B)    SLJ(2EXT)      . FULL EXIT.                        31K
* START - PLACE THE NEXT CHAR. (XLOC, YLOC) AS INDICATED.               32K
*     T - TOP OF PAGE AT LEFT MARGIN,  P - AT LOCATION OF PIPPER        32K
*     OR  Y - AT PIPPERS Y COORD. AND LEFT MARGIN.                      32K
 32A  ENA(32C)       SLJ(01AE)      .                                   32K
 32C  INA(-23B)      AJP(32T)       . T.                                32K
      INA(-5)        AJP(32Y)       . Y.                                32K
      INA(-17B)      AJP(32P)       . P.                                32K
      SLJ(1EXT)                     . NOTA.                             32K
 32T  ENA(376B)      SUB(SIZE2)     . START AT TOP.                     32K
      STA(YLOC)                     .                                   32K
 32T1 LDA(LMAR)      STA(XLOC)      . XLOC.                             32K
 32T2                SLJ4(33MK)     . POSITION MARKER.                  32K
                     SLJ4(1LOC)     . POSITION PICBUF.                  32K
      EXF(77505B)    SLJ(2EXT)      . START OFF.                        32K
 32Y  LDA(PIPY)      ARS(2)         . YLOC BY PIPPER.                   32K
      ALS(2)         STA(YLOC)      .                                   32K
      SLJ(32T1)                     .                                   32K
 32P  LDA(PIPY)      LDQ(PIPX)      . XLOC AND YLOC BY PIPPER.          32K
      ARS(2)         QRS(2)         .                                   32K
      ALS(2)         QLS(2)         .                                   32K
```

```
       STA(YLOC)      STQ(XLOC)      .                                       32K
       SLJ(32T2)                     .                                       32K
* MARKER - ANY KB2 CHAR INCLUDING SPACE.                                     33K
 33A   ENA(33C)       SLJ(01AE)      . ROUTE.                                33K
 33C   ALS(36B)                      . POSITION CHARACTER.                   33K
       LDQ(MKA)       SSU(MARKER)    .                                       33K
       STA(MARKER)    SLJ4(33MK)     . ADDRESS MARKER.                       33K
       EXF(77511B)    SLJ(2EXT)      .                                       33K
* CLEAR PICBUF.                                                              34K
 34A   LDA(NPOSIT)    STA(POSIT)     . ZEROIZE POSIT.                        34K
       ENA(0)         ENI1(1000B)    . CLEAR PICBUF.                         34K
       STA1(PICBUF)   IJP1(L)        .                                       34K
       ENI1(1000B)                   . CLEAR DD65 MEM.                       34K
       EXF(77010B)    EXF7(77010B)   . SEL AND WAIT FOR DD65 MEM.            34K
       OUT(ERC2)      IJP1(L-1)      .                                       34K
                      ENI6(1)        .                                       34K
       ENA(01)        SLJ4(1PB)      . START FILE.                           34K
       SLJ(55A)                      . LIGHTS OFF.                           34K
*  TEST KEYBOARD 2 LIGHTS.                                                   35K
 35A   EXF(77541B)    SLJ4(FLASH)    .                                       35K
       SLJ(55A)                      . EXIT VIA RESET.                       35K
* PIP ON/OFF.                                                                41K
 41A   LDA(PIPQ)      ENQ(1)         . POS = ON,  NEG = OFF.                 41K
       AJP3(L+1)      ENQ(-1)        . PIPQ NEG., TURN OFF.                  41K
       STQ(PIPQ)      SLJ(1PIP)      .     POS, TURN ON.                     41K
*  TRACK BALL.                                                               42K
 42A   EXF(77102B)    INT(PIPX)      .                                       42K
       EXF(77104B)    INT(PIPY)      .                                       42K
                      SLJ(1PIP)      .                                       42K
* CHARACTER EDIT - R=REPLACE WITH NEXT CHAR, D=DELETE AND CLOSE UP,          44K
*      I=INSERT NEXT CHAR TO THE LEFT.          FLASHING=CHAR NOT FOUND.     44K
 44A   ENA(44C)       SAU(1STP)      .                                       44K
       ENA(PICBUF)    SAL(C1+1)      .                                       44K
       ENA(POSIT)     SAL(C1+2)      .                                       44K
       ENA(HERE)      SAL(C1+3)      .                                       44K
       ENA(PIPX)      SAL(C1+4)      .                                       44K
       ENA(PIPY)      SAL(C1+5)      .                                       44K
                      SLJ4(C1)       . DO CHALOC.                            44K
       AJP2(0EXT)     ENI2(6)        .                                       44K
       EXF(77620B)    SLJ4(3WAT)     . ON.                                   44K
       EXF(77621B)    SLJ4(3WAT)     . OFF.                                  44K
       IJP2(L-2)      SLJ(0EXT)      . BLINK.                                44K
 44C   INA(-51B)      AJP(44R)       . REPLACE.                              44K
       INA(-13B)      AJP(44D)       . DELETE.                               44K
       INA(-5B)       AJP(44I)       . INSERT.                               44K
       SLJ(1EXT)                     . NOTA, ERROR EXIT.                     44K
* REPLACE CHAR WITH ANY KB1 CHAR.                                            44K
 44R   ENA(44R1)      SLJ(01AE)      . ROUTE NEXT BYTE BELOW.                44K
 44R1  LIL1(HERE)     STA(TEMP)      . CHAR TO TEMP.                         44K
```

```
      LIU2(HERE)      LDA1(PICBUF)     .                                  44K
      LRS2(0)         SCL(MK77)        . CLEAR CHAR.                      44K
      ADD(TEMP)       LLS2(0)          . INSERT NEW CHAR.                 44K
      STA1(PICBUF)                     .                                  44K
 44EX ENA(PICBUF)     SAL(PF+1)        .                                  44K
                      SLJ4(PF)         . DO PICGEN.                       44K
      EXF(77621B)     SLJ(2EXT)        . CHAR EDIT OFF.                   44K
* DELETE AND CLOSE UP.                                                    44K
 44D  LIL1(HERE)      LIU2(HERE)       .                                  44K
      LDA1(PICBUF)    LRS2(0)          . BAD BYTE IN A(RJ).               44K
      ARS(6)          LLS2(6)          . BYTE A(RJ) IS EMPTY.             44K
      STA1(PICBUF)    LDA(POSIT)       .                                  44K
      INA(1)          SAU(L+4)         . DONT PROCESS MORE THAN NEEDED.   44K
      LDA1(PICBUF)    LDQ1(PICBUF+1)   . PASS EMPTY BYTE THRU PICBUF.     44K
      ARS(6)          LLS(6)           .                                  44K
      STA1(PICBUF)    STQ1(PICBUF+1)   .                                  44K
      ISK1(N)         SLJ(L-3)         .                                  44K
      LIU1(POSIT)     ENA1(-603)       . UPDATE POSIT.                    44K
      AJP1(L+2)       RSO(POSIT)       .                                  44K
      ENI1(0)                          .                                  44K
      INI1(6)         SIU1(POSIT)      .                                  44K
      SLJ(44EX)                        . EXIT.                            44K
* INSERT NEXT KB1 CHAR                                                    44K
 44I  ENA(44I1)       SLJ(01AE)        . ROUTE.                           44K
 44I1 STA(T)          LIL1(HERE)       .                                  44K
      LIU2(HERE)      LDA1(PICBUF)     . B2=BIT WHICH DIVIDES OLD AND NEW. 44K
      LRS2(6)         ALS(6)           .                                  44K
      SCL(MK77)       ADD(T)           .                                  44K
      LLS2(0)         STA1(PICBUF)     .                                  44K
      LDA(POSIT)      INA(1)           .                                  44K
      SAU(L+3)        INI1(1)          .                                  44K
      LLS(6)          LDQ1(PICBUF)     .                                  44K
      LLS(52B)        STA1(PICBUF)     .                                  44K
      ISK1(N)         SLJ(L-2)         .                                  44K
      LIU2(POSIT)     ENA2(-6)         . UPDATE POSIT.                    44K
      AJP1(L+2)                        .                                  44K
      RAO(POSIT)      ENI2(60B)        .                                  44K
      INI2(-6)        SIU2(POSIT)      .                                  44K
      SLJ(44EX)                        . EXIT.                            44K
* LINE EDIT - R = DELETE CHAR ON PIP AND ALL TO THE RIGHT. ANY STRING OF54K
*     KB1 CHAR CAN FOLLOW (EXCEPT TAB OR CR) UNTILL END OF                54K
*     STRING IS REACHED, WENCE AN AUTO EXIT OCCURES. TAB IS IGNORED AND 54K
*     CR CAUSES ANIMEDIATE EXIT.   FLASHING KEY = LINE CANT BE FOUND.    54K
*           - D = DELETE AND CLOSE UP, SILF EXIT.                         54K
*           - I = INSERT LINE (NOT PROGRAMED).                            54K
 54A  ENA(54C)        SAU(1STP)        . ROUTE NEXT HIT.                  54K
      ENA(PICBUF)     SAL(C1+1)        .                                  54K
      ENA(POSIT)      SAL(C1+2)        .                                  54K
      ENA(HERE)       SAL(C1+3)        .                                  54K
```

```
        ENA(PIPX)       SAL(C1+4)       .                                         54K
        ENA(PIPY)       SAL(C1+5)       .                                         54K
                        SLJ4(C1)        . DO CHALOC.                              54K
        AJP2(0EXT)      ENI2(6)         . NOT HERE, BLINK KEY.                    54K
        EXF(77720B)     SLJ4(3WAI)      . ON.                                     54K
        EXF(77721B)     SLJ4(3WAI)      . OFF.                                    54K
        IJP2(L-2)       SLJ(2EXT)       .                                         54K
 54C    INA(-51B)       AJP(54R)        . DELETE AND REPLACE.                     54K
        INA(-13B)       AJP(54D)        . DELETE AND CLOSE UP.                    54K
        SLJ(1EXT)                       . NOTA.                                   54K
*  REPLACE.                                                                       54K
 54R    ENA(L+1)        SLJ(01AE)       . ROUTE NEXT HIT BELOW.                   54K
        LIL1(HERE)      LIU2(HERE)      . RELOAD INDICIES.                        54K
        AJP(0EXT)       INA(-36B)       . IGNORE BACKSPACE AND TAB.               54K
        AJP(0EXT)       INA(-40B)       .                                         54K
        AJP2(54RX)      INA(76B)        . EXIT ON CR.                             54K
        STA(T)          LDA1(PICBUF)    .                                         54K
        LRS2(6)         ENA(77777B)     . OLD BYTE IN Q(LJ).                      54K
        LLS(6)          AJP(54RX)       . END OF STRING, EXIT.                    54K
        LDA(T)          LRS(6)          . NEW BYTE IN Q(LJ).                      54K
        LDA1(PICBUF)    ARS2(6)         .                                         54K
        LLS2(6)         STA1(PICBUF)    . NEW WORD FORMED AND STORED.             54K
        ENA(PICBUF)     SAL(PF+1)       . PRINT                                   54K
                        SLJ4(PF)        .                                         54K
        IJP2(L+2)                       . NOT LAST BYTE IN WORD, JJMP.            54K
        RAO(HERE)       ENI2(57B)       . CALL NEXT WORD.                         54K
        INI2(-5)        SIU2(HERE)      . STORE INDICIES.                         54K
        SLJ(0EXT)                       .                                         54K
 54RX   EXF(77721B)     SLJ(2EXT)       . LIGHT OFF AND EXIT.                     54K
*  DELETE AND CLOSE UP.                                                           54K
 54D    LIL1(HERE)      LIU2(HERE)      . FIND THE START OF THE STRING(00).       54K
        LDA1(PICBUF)    LRS2(6)         . X BYTE IN Q(LJ).                        54K
        ENQ(77777B)     LLS2(6)         . X BYTE AND ALL TO THE RIGHT = 77.       54K
        ENI3(7)                         . ***                                     54K
        ENQ(0)          LRS(6)          . * *** EXAM BYTES FROM RT TO LF.         54K
        QJP(54DF)       IJP3(L-1)       . * ***  00 FOUND, JUMP.                  54K
        IJP1(L+1)       SLJ(2EXT)       . *  PICBUF EXAUSTED, JUMP.               54K
        LDA1(PICBUF)    SLJ(L-4)        . ***                                     54K
 54DF   ENI4(10B)                       .                                         54K
        INI4(-1)        IJP3(L)         . B4 = 7 - B3.                            54K
        LDQ1(PICBUF)                    . ***  REPLACE ALL BYTES IN THE           54K
        LDA4(K0)        SBL4(K0)        . * ***  STRING WITH 77.                  54K
        AJP(54DA)       LDA1(PICBUF)    . * *  END OF STRING, JUMP.               54K
        SST4(K0)        STA1(PICBUF)    . * *                                     54K
        IJP4(L-3)       ENI4(7)         . * ***                                   54K
        ISK1(1000B)     SLJ(L-5)        . ***                                     54K
*  MOVE EACH LINE UP ONE.                                                         54K
 54DA   LIU2(HERE)      LIL1(HERE)      .                                         54K
        LDA1(PICBUF)    LRS2(0)         .                                         54K
```

```
      STQ(W)          ENA2(0)        .                                    54K
      ENQ(0)                         .                                    54K
      DVI(M6)         SAU(L+1)       .                                    54K
      ENI2(N)         LDA(PIPY)      . CAL PERMINENT NEW Y.               54K
      ARS(2)          AJP2(L+2)      . PIPY TO PICGEN CODE.               54K
      SCM(SEV)        INA(200B)      .                                    54K
      INA(2100B)      STA(PNY)       .                                    54K
      STA(TNY)                       .                                    54K
 54D1 ENA(54DC)                      .                                    54K
      SAU(54DB)                      .                                    54K
*  READ NEXT BYTE.                                                        54K
 54DR IJP2(L+4)       ENI2(7)        .                                    54K
      ISK1(1000B)     SLJ(L+2)       .                                    54K
      SLJ(54DE)                      . PICBUF EXAUSTED, EXIT.             54K
      LDA1(PICBUF)    STA(W)         .                                    54K
      LDQ(W)          ENA(0)         .                                    54K
      LLS(6)          STQ(W)         .                                    54K
*  BRANCH POINT.                                                          54K
 54DB SLJ(N)                         .                                    54K
*  SKIP STRING.                                                           54K
 54DS INA(-77B)       AJP1(54DR)     . NOT END OF STRING, EXIT.           54K
      SLJ(54D1)                      .                                    54K
*  DECODER.                                                               54K
 54DC AJP(54D0)       INA(-20B)      . BEGIN STRING, JUMP.                54K
      AJP3(54DR)      INA(-4)        . LESS THAN 20, READ NEXT BYTE.      54K
      AJP2(L+2)       LRS(1)         . NOT A LOC, JUMP.                   54K
      QJP3(54DX)      SLJ(54DY)      . 20 + 22 TO X,  21 + 23 TO Y.       54K
      INA(-53B)       AJP1(54DR)     . NOT END OF FILE, EXIT.             54K
      SLJ(54DR+2)                    . FINAL EXIT.                        54K
*  PROCESS BEGIN STRING.                                                  54K
 54D0 LDA(TNY)        STA(PNY)       . TEMP BECOMES PERM YLOC.            54K
      ENA(54DS)       SLJ(54D1+1)    . SKIP STRING.                       54K
*  PROCESS X LOCATION.                                                    54K
 54DX ENA(L+1)        SLJ(54D1+1)    . SKIP NEXT BYTE.                    54K
      SLJ(54D1)                      . RESTORE NORMAL ROUTE.              54K
*  PROCESS Y LOCATION.                                                    54K
 54DY LLS(1)          INA(24B)       . EXTRACT THIS Y FOR USE ON NEXT     54K
      ALS(6)          STA(TNY)       .     LINE.                          54K
      ENA(L+1)        SLJ(54D1+1)    .                                    54K
      RAD(TNY)        ENA2(-7)       .                                    54K
      AJP(L+6)        INA(7)         .                                    54K
      MUI(M6)         SAU(L+2)       .                                    54K
      SAU(L+2)        LDA(PNY)       .                                    54K
      ENQ(77000B)     ALS(N)         .                                    54K
      QLS(N)          SSU1(PICBUF)   .                                    54K
      STA1(PICBUF)    SLJ(54D1)      .                                    54K
      LDQ1(PICBUF)    LLS(6)         .                                    54K
      LDA(PNY)        LRS(6)         .                                    54K
      STQ1(PICBUF)    LRS(6)         .                                    54K
```

```
      INI1(-1)                                  .                                   54K
      LDA1(PICBUF)    ARS(6)                    .                                   54K
      LLS(6)          STA1(PICBUF)              .                                   54K
      INI1(1)         SLJ(54DR)                 .                                   54K
 54DE ENA(PICBUF)     SAL(PF+1)                 . PRINT AND EXIT.                   54K
                      SLJ4(PF)                  .                                   54K
      EXF(77721B)     SLJ(2EXT)                 .                                   54K
* RESET                                                                             55K
 55A  EXF(77203B)     EXF(77205B)               . ALL LIGHTS OUT.                   55K
      EXF(77211B)     EXF(77221B)               .                                   55K
      EXF(77241B)     EXF(77303B)               .                                   55K
      EXF(77305B)     EXF(77311B)               .                                   55K
      EXF(77321B)     EXF(77341B)               .                                   55K
      EXF(77403B)     EXF(77405B)               .                                   55K
      EXF(77411B)     EXF(77421B)               .                                   55K
      EXF(77441B)     EXF(77503B)               .                                   55K
      EXF(77505B)     EXF(77511B)               .                                   55K
      EXF(77521B)     EXF(77541B)               .                                   55K
      EXF(77603B)     EXF(77605B)               .                                   55K
      EXF(77611B)     EXF(77621B)               .                                   55K
                      EXF(77703B)               .                                   55K
      EXF(77705B)     EXF(77711B)               .                                   55K
      EXF(77721B)     ENI6(1)                   .     EXCEPT RESET.                 55K
      ENA(77B)        SLJ4(1PB)                 . END STRING IN PICBUF.             55K
      SLJ(2EXT)                                 .                                   55K
* PIP UP.                                                                           61K
 61A  LDA(PIPD)       INA(-1)                   . PIPD = LAST 6X BUTTON PUSHED.     61K
      AJP(L+3)        ENA(1)                    .    I.E. 1=61, 2=62, 3=63, 0=64.   61K
      STA( 'IPD)      ENA(0)                    .                                   61K
      STA(TIM)        SLJ(2EXT)                 .                                   61K
      RAO(TIM)        RAD(PIPY)                 .  PIPX AND PIPY = PIP LOC (512).   61K
      SLJ(1PIP)                                 .                                   61K
* PIP RIGHT.                                                                        62K
 62A  LDA(PIPD)       INA(-2)                   . RIGHT.                            62K
      AJP(L+3)        ENA(2)                    .                                   62K
      STA(PIPD)       ENA(0)                    .                                   62K
      STA(TIM)        SLJ(2EXT)                 .                                   62K
      RAO(TIM)        RAD(PIPX)                 .                                   62K
      SLJ(1PIP)                                 .                                   62K
* PIP DOWN.                                                                         63K
 63A  LDA(PIPD)       INA(-3)                   .  DOWN.                            63K
      AJP(L+3)        ENA(3)                    .                                   63K
      STA(PIPD)       ENA(0)                    .                                   63K
      STA(TIM)        SLJ(2EXT)                 .                                   63K
      RAO(TIM)        RSB(PIPY)                 .                                   63K
      SLJ(1PIP)                                 .                                   63K
* PIP LEFT.                                                                         64K
 64A  LDA(PIPD)       AJP(L+3)                  . LEFT.                             64K
      ENA(0)          STA(PIPD)                 .                                   64K
```

```
      STA(TIM)       SLJ(2EXT)      .                                   64K
      RAO(TIM)       RSB(PIPX)      .                                   64K
      SLJ(1PIP)                     .                                   64K


* INTER-KEY SUBROUTINES.                                                SRK

* PRINT TAB LIST.                                                       PTK
 1PTL ENI1(0)        LDQ(MKX)       . LOAD  X POSIT MASK                PTK
      LDA1(TL1)      ALS(44B)       . POSITION X                        PTK
      SSU1(TB1)      STA(T)         .                                   PTK
      EXF(77010B)    EXF7(77010B)   . SEL AND WAIT FOR DD65 MEM.        PTK
                     OUT(T)         . OUTPUT.                           PTK
      ISK1(17B)      SLJ(L-4)       .                                   PTK
      EXF(77311B)    EXF(77321B)    . LIGHTS OUT.                       PTK
      SLJ(2EXT)                     .                                   PTK
* CAL DELTA.                                                            DLK
 15Z  SLJ(N)         LDA(SPAC)      . DELTA = SPACING * SIZE.           DLK
      MUI(SIZE)      STA(DELTA)     .                                   DLK
      SLJ(L-2)                      .                                   DLK
* PIP INCREMENTATION.                                                   PIK
 1PIP LDA(PIPQ)      AJP3(2PIP)     . IF PIP IS OFF, JUMP.              PIK
      LDA(PIPX)      LDQ(PIPY)      . EXTEND SIGN.                      PIK
      LLS(47B)       ARS(47B)       .                                   PIK
      QRS(47B)       STA(PIPX)      .                                   PIK
      STQ(PIPY)      QLS(14B)       . POSITION ADDRESSES.               PIK
      LLS(44B)       LDQ(MKXY)      .                                   PIK
      SSU(PIP)       STA(PIP)       . INSERT NEW ADDRESSES.             PIK
      EXF(77010B)    EXF7(77010B)   . SEL AND WAIT FOR DD65 MEM.        PIK
                     OUT(PIP)       . PRINT.                            PIK
      EXF7(77174B)   SLJ(L+3)       . KB2 HIT,JUMP.                     PIK
      ENI1(1)        ISK1(10000B)   . WAIT.                             PIK
      EXF7(77175B)   SLJ(2EXT)      . STILL NOT HIT, RETURN.            PIK
      EXF(77120B)    INT(T)         . DUMP INPUT.                       PIK
      SLJ(L-4)                      . LOOP.                             PIK
 2PIP EXF(77010B)    EXF7(77010B)   . SEL AND WAIT FOR DD65 MEM.        PIK
                     OUT(PIPOFF)    . TURN OFF PIP (KB2).               PIK
      EXF(77603B)    SLJ(2EXT)      .                                   PIK
* CONVERT A SIGNED 9 BIT RJ NR TO A SIGN + 2 BYTE BCD OCT NR.           VTK
 1VRT SLJ(N)         ENQ(6000B)     . BCD +.                            VTK
      AJP2(L+2)      SCM(SEV)       . IF NEG, COMPLEMENT.               VTK
      ENQ(4000B)                    . BCD -.                            VTK
      STQ(TEMP)      LRS(10B)       .                                   VTK
      ENA(0)         LLS(3)         .                                   VTK
      RAD(TEMP)      ENA(0)         .                                   VTK
      LLS(3)         AJP1(L+2)      .                                   VTK
      ENA(12B)                      .                                   VTK
      LRS(6)         LDA(TEMP)      .                                   VTK
```

```
      LLS(6)          SLJ(1VRT)       .                                    VTK
* PLACE XLOC AND YLOC IN PICBUF.                                           LCK
 1LOC SLJ(N)          ENI6(10B)       . DONT PRINT.                        LCK
      LDA(XLOC)       LDQ(YLOC)       . EXTEND SIGN.                       LCK
      LLS(47B)        ARS(47B)        .                                    LCK
      QRS(47B)        STA(XLOC)       .                                    LCK
      STQ(YLOC)       AJP2(L+3)       . XLOC POS., JUMP.                   LCK
      ENA(22B)        SLJ4(1PB)       . -X = NEXT BYTE.                    LCK
      LAC(XLOC)       SLJ(L+3)        .                                    LCK
      ENA(20B)        SLJ4(1PB)       . +X = NEXT BYTE.                    LCK
      LDA(XLOC)                       .                                    LCK
      ARS(2)          SLJ4(1PB)       . POSITION 2 BYTES.                  LCK
      LDA(YLOC)       AJP2(L+3)       . START YLOC LIKE XLOC.              LCK
      ENA(23B)        SLJ4(1PB)       .                                    LCK
      LAC(YLOC)       SLJ(L+3)        .                                    LCK
      ENA(21B)        SLJ4(1PB)       .                                    LCK
      LDA(YLOC)                       .                                    LCK
      ARS(2)          SLJ4(1PB)       . POSITION 2 BYTES.                  LCK
      SLJ(1LOC)                       .                                    LCK
* LOADER FOR PICBUF.  B6 = 0-CALL PICGEN, NOT = 0-SKIP IT.                 PBK
 1PB  SLJ(N)          SIL1(6PB)       .                                    PBK
      SIU2(6PB)       LIL1(POSIT)     . B1=PICBUF WORD TO BE ADDED TO.     PBK
      LIU2(POSIT)     LDQ(TERM)       . B2= UNUSED BITS IN PICBUF(B1).     PBK
      LRS(6)          LDA1(PICBUF)    . Q(LS)  = NEW BYTE + 77 + 76.       PBK
      ARS2(0)         LLS2(0)         . FORM NEW WORD.                     PBK
      STA1(PICBUF)    STQ1(PICBUF+1)  . PICBUF IS PACKED.                  PBK
      ENA2(-6)        AJP1(4PB)       . NEW WORD NOT NEEDED, JUMP.         PBK
      ISK1(777B)      SLJ(2PB)        . PICBUF NOT FULL, JUMP.             PBK
      ENI6(3)         EXF(77010B)     . PRINT PICBUF IS FULL.              PBK
      OUT6(FUL3)                      .                                    PBK
 2PB  ENI2(66B)                       . NEW WORD, ALL 48 BITS UNUSED.      PBK
 4PB  INI2(-6)        IJP6(3PB)       . B6 = 0, SKIP PICGEN                PBK
      ENA(PICBUF)     SAL(PF+1)       .                                    PBK
                      SLJ4(PF)        . DO PICGEN.                         PBK
 3PB  SIL1(POSIT)     SIU2(POSIT)     .                                    PBK
 6PB  ENI2(N)         ENI1(N)         .                                    PBK
      ENA(0)          SLJ(1PB)        .                                    PBK
* WAIT.                                                                    WTK
 1WAT SLJ(N)          ENI1(40B)       . SHORT WAIT.                        WTK
      IJP1(L)         SLJ(L-1)        .                                    WTK
 3WAT SLJ(N)          ENA(4)          . .3 TO .4 SEC WAIT.                 WTK
      AJP(L-1)        ISK1(30000B)    .                                    WTK
      INA(-1)         SLJ(L-1)        .                                    WTK
* PRINT LEFT AND RIGHT MARGINS.                                            MRK
 1MAR SLJ(N)          LDA(RMAR)       .                                    MRK
      INA(3)          ALS(44B)        . X TO RIGHT MARGIN.                 MRK
      LDQ(MKX)        SSU(RM1)        .                                    MRK
      STA(RM1)        LDA(LMAR)       .                                    MRK
      INA(-3)         ALS(44B)        . X TO LEFT MARGIN.                  MRK
```

```
     SSU(LM1)    STA(LM1)     .                             MRK
     LDA(RMAR)   SLJ4(1VRT)   .                             MRK
     ALS(14B)    LDQ(MKN)     .                             MRK
     SSU(RM2)    STA(RM2)     . X TO RM2.                   MRK
     LDA(LMAR)   SLJ4(1VRT)   .                             MRK
     ALS(14B)    LDQ(MKN)     .                             MRK
     SSU(LM2)    STA(LM2)     . X TO LM2.                   MRK
     ENI1(4)                  .                             MRK
     EXF(77010B) EXF7(77010B) . SEL AND WAIT FOR DD65 MEM.  MRK
     OUT1(RM2)   LDA(RMAR)    .                             MRK
     SUB(LMAR)   ENQ(0)       .                             MRK
     AJP2(L+1)   ENA(0)       . NO NEG.                     MRK
     DVI(SIZE)   INA(-144B)   . GREATER THAN 100            MRK
     AJP2(L+2)   INA(144B)    .     YES, JUMP.              MRK
     ENQ(0)      SLJ(L+2)     .                             MRK
     ENQ(100B)                .                             MRK
     STQ(T)      ENI1(0)      . HOW MANY TENS.              MRK
     INA(-12B)   AJP3(L+2)    .                             MRK
     INI1(1)     SLJ(L-1)     .                             MRK
     INA(12B)    AJP1(L+2)    .                             MRK
     ENA(12B)                 . ZERO = 12 IN BCD.           MRK
     LRS(6)      ENA1(0)      .                             MRK
     AJP1(L+1)   ENA(12B)     .                             MRK
     ADD(T)      LLS(36B)     .                             MRK
     LDQ(MKA)                 .                             MRK
     SSU(CPL3)   STA(CPL3)    .                             MRK
     ENI1(3)                  . OUTPUT TO DESPLAY.          MRK
     EXF(77010B) EXF7(77010B) . SEL AND WAIT FOR DD65 MEM.  MRK
     OUT1(CPL3)  SLJ(1MAR)    .                             MRK
* NEXT 2 HITS CONVERTED TO BINARY.                          RTK
 1RET SLJ(N)     INA(-12B)    .                             RTK
     AJP(L+2)    INA(2)       . BCD 0, JUMP.                RTK
     AJP2(1EXT)  INA(10B)     . NON-OCTAL, EXIT.            RTK
     ALS(5)      STA(T)       . POSITION 1ST BYTE.          RTK
     ENA(L+1)    SLJ(01AE)    . ROUTE LAST HIT.             RTK
     INA(-12B)   AJP(L+3)     .                             RTK
     INA(2)      AJP(1EXT)    .                             RTK
     INA(10B)    ALS(2)       .                             RTK
     ADD(T)      SCL(MKY)     .                             RTK
                 SLJ(1RET)    .                             RTK
* PRINT MARKER.                                             MKK
 33MK SLJ(N)     LDA(XLOC)    .                             MKK
     LDQ(YLOC)   QLS(14B)     .                             MKK
     LLS(44B)    LDQ(MKXY)    . POSITION X + Y. LOAD MASK.  MKK
     SSU(MARKER) STA(MARKER)  .                             MKK
     EXF(77010B) EXF7(77010B) . SEL AND WAIT FOR DD65 MEM.  MKK
     OUT(MARKER) SLJ(33MK)    .                             MKK
* BACKSPACE                                                 BSK
 1BKS            LDA(SIZE)    . BACKSPACE MARKER.           BSK
```

```
      SSK(INCR)      SLJ(L+2)       .                                  BSK
      RSB(XLOC)      SLJ(L+2)       .                                  BSK
      RAD(YLOC)                     .                                  BSK
                     SLJ4(33MK)     .                                  BSK
      LIL1(POSIT)    LIU2(POSIT)    . BACKSPACE PICBUF.                BSK
      ENA2(-45B)     AJP3(L+2)      .                                  BSK
      INI1(-1)       INI2(-60B)     .                                  BSK
      INI2(14B)      SIL1(POSIT)    .                                  BSK
      SIU2(POSIT)    ENI6(0)        . PRINT NEW PICBUF.                BSK
      LDA1(PICBUF)   LRS2(0)        .                                  BSK
      ENA(0)         LLS(6)         .                                  BSK
      AJP5(1PB)      SLJ(1EXT)      .                                  BSK
      SLJ(0EXT)                     .                                  BSK
      END



      MACHINE PICGEN(2ST)                                              PRP
*  B1 = NUMBER OF BYTES LEFT IN PICBUF(B2).      B5 = TEMP STORAGE.    PRP
*  B3 = NEXT MEMORY ADDRESS.  B4 = BYTES TILL OUTPUT WORD.             PRP
      CON(MKSPT=0000 7320 0000 0000B, MKCH =0000 0000 0000 2000B,      PRP
     1    MKCM =6000 0000 0000 0177B, MKX  =7000 7777 7777 7777B,      PRP
     2    MKY  =7777 7777 7777 7000B, MKPDW=7777 0000 7777 7777B,      PRP
     3    MKADD=7777 7777 6001 7777B, MK47 =4000 0000 0000 0000B,      PRP
     4    MK46 =2000 0000 0000 0000B, MK45 =1000 0000 0000 0000B,      PRP
     5    MK23 =0000 0000 4000 0000B, MK11 =0000 0000 0000 4000B,      PRP
     6    MKXY =7000 7777 7777 7000B)                                  PRP
      CON(ERR4 =6346 6465 7320 4020B, ERR3 =2071 4547 2423 4020B,      PRP
     1    ERR2 =5146 4565 4624 4022B, ERR1 =5567 6551 1652 5000B,      PRP
     2    ER2  =0605 2066 2443 4043B, ER1  =5667 6464 1660 5000B)      PRP
 1ST  SLJ(N)         SLJ(L+2)       .                                  STP
 2ST  ZRO(0)         ZRO(0)         . (PICBUF).                        STP
      LDA(2ST)       SAU(2RNB)      . LOCATE PICBUF.                   STP
      SIU1(1FIN)     SIL2(1FIN)     . SAVE INDICIES.                   STP
      SIU3(2FIN)     SIL4(2FIN)     .                                  STP
      SIU5(3FIN)     SIL6(3FIN)     .                                  STP
      ENA(0)         STA(W)         . CLEAR W.                         STP
      ENI1(0)        ENI2(0)        . SET INDICES.                     STP
      ENI4(1)        ENI5(2)        .                                  STP
      ENA(4PAK)      SAL(2PAK)      .                                  STP
      ENA(L+1)       SLJ(4ST)       . WAIT FOR START FILE.             STP
      INA(-1)        AJP1(1RNB)     . PRIME DW AND SW.                 STP
      STA(SW)        ENA(1000B)     . INITIAL DW.                      STP
      STA(DW)        ENI3(0)        . PRIME MEM. ADDRESS.              STP
 3ST  ENA(1DG)                      . SET ROUTE FOR NON-STRING.        STP
 4ST  SAU(1BRA)                     . ALTER ROUTE.                     STP
* READ NEXT BYTE.                                                      RBP
```

```
1RNB IJP1(L+5)       ENI1(7B)          . READ NEXT BYTE.                      RBP
     ISK2(1000B)     SLJ(2RNB)         . ANY WORDS LEFT.                      RBP
     SLJ(1FN)                          . NO, EXIT.                            RBP
     ZRO(0)          ZRO(0)            . WORD                                 RBP
2RNB LDA2(N)         STA(L-1)          . LOAD NEW WORD.                       RBP
     LDQ(L-2)        ENA(0)            . WORD TO Q, 0 TO A.                   RBP
     LLS(6B)         STQ(L-3)          . LOAD NEXT BYTE, Q TO WORD.           RBP
* MAIN BRANCH POINT.                                                          BPP
1BRA SLJ(N)                            . MAIN BRANCH POINT.                   BPP
* PROCESS STRING.                                                             PSP
1CST INA(-76B)       AJP2(1EST)        . IF BYTE IS CR OR END STRING, JUMP.PSP
     INA(76B)        SLJ4(1PAK)        . PACK.                                PSP
     SLJ(1RNB)                         .                                      PSP
1SPT ALS(6)          STA(W1)           . 1ST BYTE.                            PSP
     ENA(L+1)        SLJ(4ST)          .                                      PSP
     ADD(W1)         ALS(6)            . 2ND BYTE.                            PSP
     STA(W1)                           .                                      PSP
     ENA(L+1)        SLJ(4ST)          .                                      PSP
     ADD(W1)         LRS(11B)          . 3RD AND LAST BYTE.                   PSP
     ALS(33B)        LLS(11B)          . COOR. ARE IN POSITION.               PSP
     LDQ(MKXY)       SSU(DW)           . INSERT LOCATION.                     PSP
     STA(W1)         LDQ(MKPDW)        .                                      PSP
     LDA(MKSPT)      SSU(W1)           . INSERT DOT.                          PSP
     STA(W1)         INI3(1)           . INDIX MEM.                           PSP
     EXF(77010B)     EXF7(77010B)      . SEL AND WAIT FOR DD65 MEM.           PSP
     OUT(W1)         SLJ(3ST)          . OUTPUT.                              PSP
* END STRING.                                                                 ESP
1EST ENA4(-6)        AJP(3ST)          . OUTPUT WORD COMPLETE, JUMP.          ESP
     ENA(20B)        SLJ4(1PAK)        . FILL WITH BLANKS.                    ESP
2EST SLJ(1EST)                         .                                      ESP
* PACK                                                                        PKP
1PAK SLJ(N)          RAD(W)            .                                      PKP
2PAK IJP4(L+1)       SLJ(N)            . WORD FULL, JUMP TO 4 OR 5 PAK        PKP
     ALS(6)          STA(W)            . NOT FULL, STORE AND RETURN.          PKP
     SLJ(1PAK)                         .                                      PKP
4PAK ENA(5PAK)       SAL(2PAK)         . DW.                                  PKP
     LDA(W)                            .                                      PKP
     ALS(30B)        LDQ(MKPDW)        .                                      PKP
     SSU(DW)         SLJ(L+4)          .                                      PKP
5PAK LRS(6B)         ALS(6B)           . SW STRING.                           PKP
     LLS(6B)         ENQ(07700B)       .                                      PKP
     SSU(SW)                           .                                      PKP
     ISK3(727B)      SLJ(L+2)          . UPDATE MEM ADDRESS.                  PKP
     SLJ(1ER)                          . AVAILABLE MEM FILLED, EXIT.          PKP
     EXF(77010B)     EXF7(77010B)      . SEL AND WAIT FOR DD65 MEM.           PKP
     ENI4(6)         STA(W)            . NEW WORD.                            PKP
     OUT(W)          ENA(0)            .                                      PKP
     STA(W)          SLJ(1PAK)         .                                      PKP
* TABLE                                                                       TBP
```

```
1DG  INA(00D)     SAU(L+1)     . ENTER TABLE.                TBP
     SLJ(N)                    .    JUMP.                    TBP
00D  ENA3(0)      SLJ(1BGS)    . BEGIN STRING.               TBP
     SLJ(1RNB)                 . PASS.                       TBP
     ENA(2ERR)    SLJ(1ERR)    .                             TBP
     ENA(2ERR)    SLJ(1ERR)    .                             TBP
     ENA(2ERR)    SLJ(1ERR)    .                             TBP
     ENA(2ERR)    SLJ(1ERR)    .                             TBP
     ENA(2ERR)    SLJ(1ERR)    .                             TBP
     ENA(2ERR)    SLJ(1ERR)    .                             TBP
10D  ENA(1CST)    SLJ(1CH)     . CHARACTER STRING.           TBP
     ENA(1CST)    SLJ(1VCT)    . VECTOR STRING.              TBP
     ENA(1SPT)    SLJ(1CH)     . SPOT MODE.                  TBP
     ENA(2ERR)    SLJ(1ERR)    .                             TBP
     ENA(2ERR)    SLJ(1ERR)    .                             TBP
     ENA(2ERR)    SLJ(1ERR)    .                             TBP
     ENA(2ERR)    SLJ(1ERR)    .                             TBP
     ENA(2ERR)    SLJ(1ERR)    .                             TBP
20D  ENA(20AD)    SLJ(4ST)     . SET X.                      TBP
     ENA(21AD)    SLJ(4ST)     . SET Y.                      TBP
     ENA(22AD)    SLJ(4ST)     . SET -X.                     TBP
     ENA(23AD)    SLJ(4ST)     . SET-Y.                      TBP
     ENA(2ERR)    SLJ(1ERR)    .                             TBP
     ENA(2ERR)    SLJ(1ERR)    .                             TBP
     ENA(2ERR)    SLJ(1ERR)    .                             TBP
     ENA(2ERR)    SLJ(1ERR)    .                             TBP
30D  LDA(DW)      SLJ(1SML)    . SET SMALL CHAR.             TBP
     LDA(DW)      SLJ(1MED)    . SET MEDIUM CHAR.            TBP
     LDA(DW)      SLJ(1LRG)    . SET LARGE CHAR.             TBP
     ENA(2ERR)    SLJ(1ERR)    .                             TBP
     ENA(2ERR)    SLJ(1ERR)    .                             TBP
     ENA(2ERR)    SLJ(1ERR)    .                             TBP
     ENA(2ERR)    SLJ(1ERR)    .                             TBP
     ENA(2ERR)    SLJ(1ERR)    .                             TBP
40D  LDA(DW)      SLJ(1TRT)    . USE RIGHT TUBE.             TBP
     LDA(DW)      SLJ(1TLF)    . USE LEFT TUBE.              TBP
     ENA(2ERR)    SLJ(1ERR)    .                             TBP
     ENA(2ERR)    SLJ(1ERR)    .                             TBP
     ENA(2ERR)    SLJ(1ERR)    .                             TBP
     ENA(2ERR)    SLJ(1ERR)    .                             TBP
     ENA(2ERR)    SLJ(1ERR)    .                             TBP
     ENA(2ERR)    SLJ(1ERR)    .                             TBP
50D  LDA(DW)      SLJ(1NOR)    . SET NORMAL INTENSITY.       TBP
     LDA(DW)      SLJ(1BRT)    . SET BRIGHT INTENSITY.       TBP
     ENA(2ERR)    SLJ(1ERR)    .                             TBP
     ENA(2ERR)    SLJ(1ERR)    .                             TBP
     ENA(2ERR)    SLJ(1ERR)    .                             TBP
     ENA(2ERR)    SLJ(1ERR)    .                             TBP
     ENA(2ERR)    SLJ(1ERR)    .                             TBP
```

```
      ENA(2ERR)       SLJ(1ERR)       .                                TBP
 60D  LDA(DW)         SLJ(1INR)       . INCREMENT RIGHT.               TBP
      LDA(DW)         SLJ(1IND)       . INCREMENT DOWN.                TBP
      ENA(2ERR)       SLJ(1ERR)       .                                TBP
      ENA(2ERR)       SLJ(1ERR)       .                                TBP
      ENA(2ERR)       SLJ(1ERR)       .                                TBP
      ENA(2ERR)       SLJ(1ERR)       .                                TBP
      ENA(2ERR)       SLJ(1ERR)       .                                TBP
      ENA(2ERR)       SLJ(1ERR)       .                                TBP
      ENA(2ERR)       SLJ(1ERR)       .                                TBP
      ENA(2ERR)       SLJ(1ERR)       .                                TBP
      ENA(2ERR)       SLJ(1ERR)       .                                TBP
      ENA(2ERR)       SLJ(1ERR)       .                                TBP
      ENA(2ERR)       SLJ(1ERR)       .                                TBP
      ENA(2ERR)       SLJ(1ERR)       .                                TBP
 76D  SLJ(1FN)                        . EXIT. NEXT DD65 ADD IN UA OF ARG. TBP
 77D  SLJ(1RNB)                       . SKIP ONE BYTE.                 TBP
* BEGIN STRING.                                                        BSP
 1BGS ALS(15B)        LDQ(MKADD)      . BEGIN STRING.                  BSP
      SSU(DW)         STA(DW)         .                                BSP
      ENA(4PAK)       SAL(2PAK)       .                                BSP
 2BGS ENI4(1)         ENA(1CST)       .                                BSP
      SLJ(4ST)                        .                                BSP
* MODE.                                                                MDP
 1CH  SAL(2BGS)       LDA(DW)         . SET CHAR. MODE.                MDP
      SCL(MKCH)       STA(DW)         .                                MDP
      LDA(SW)         SCL(MKCH)       .                                MDP
      STA(SW)         SLJ(1RNB)       .                                MDP
 1VCT SAL(2BGS)       LDA(DW)         . SET VECTOR MODE.               MDP
      SST(MKCH)       STA(DW)         .                                MDP
      LDA(SW)         SST(MKCH)       .                                MDP
      STA(SW)         SLJ(1RNB)       .                                MDP
* LOCATION.                                                            LCP
 22AD SCM(MKCM)                       . -A = X.                        LCP
 20AD ALS(46B)        LDQ(MKX)        . A = X.                         LCP
      SSU(DW)         SLJ(L+4)        .                                LCP
 23AD SCM(MKCM)                       . -A = Y.                        LCP
 21AD ALS(2B)         LDQ(MKY)        . A = Y.                         LCP
      SSU(DW)                         .                                LCP
      STA(DW)         SLJ(3ST)        . RESTORE ROUTE.                 LCP
* SIZE.                                                                SZP
 1SML SCL(MK47)       SCL(MK46)       . SET FOR SMALL.                 SZP
      STA(DW)         SLJ(1RNB)       .                                SZP
 1MED SCL(MK47)       SST(MK46)       . SET FOR MEDIUM.                SZP
      STA(DW)         SLJ(1RNB)       .                                SZP
 1LRG SST(MK47)       SCL(MK46)       . SET FOR LARGE.                 SZP
      STA(DW)         SLJ(1RNB)       .                                SZP
* TUBE.                                                                TBP
 1TRT SST(MK11)       STA(DW)         . SET RIGHT TUBE.                TBP
```

```
      LDA(SW)       SST(MK11)     .                                   TBP
      STA(SW)       SLJ(1RNB)     .                                   TBP
 1TLF SCL(MK11)     STA(DW)       . SET LEFT TUBE.                    TBP
      LDA(SW)       SCL(MK11)     .                                   TBP
      STA(SW)       SLJ(1RNB)     .                                   TBP
* INTENSITY.                                                          ITP
 1NOR SCL(MK45)     STA(DW)       . SET NORMAL INTENSITY.             ITP
      SLJ(1RNB)                   .                                   ITP
 1BRT SST(MK45)     STA(DW)       . SET BRIGHT INTENSEITY.            ITP
      SLJ(1RNB)                   .                                   ITP
* INCREMENT.                                                          ICP
 1INR SCL(MK23)     STA(DW)       . SET TO INCREMENT RIGHT.           ICP
      SLJ(1RNB)                   .                                   ICP
 1IND SST(MK23)     STA(DW)       . SET TO INCREMENT DOWN.            ICP
      SLJ(1RNB)                   .                                   ICP
* ERROR.                                                              ERP
 1ERR INI5(4)       EXF(77010B)   . PRINT ERROR.                      ERP
 2ERR OUT5(ERR4)    SLJ(1FN)      . BAD INPUT TO PICGEN.              ERP
 1ER  EXF(77010B)   EXF7(77010B)  . SEL AND WAIT FOR DD65 MEM.        ERP
      ENI1(2)       OUT1(ER2)     . PRINT ERROR.                      ERP
* END                                                                 FNP
 1FN  ENA(L+2)      SAL(1EST)     . RETURN TO HERE.                   FNP
      SLJ(1EST)                   . END STRING.                       FNP
      ENA(3ST)      SAL(1EST)     . RESTORE ROUTE.                    FNP
 1FIN ENI1(N)       ENI2(N)       .                                   FNP
 2FIN ENI3(N)       ENI4(N)       .                                   FNP
 3FIN ENI5(N)       ENI6(N)       .                                   FNP
      SLJ(1ST)                    . EXIT.                             FNP
      END



      MACHINE CHALOC (PICBUF, POSIT, HERE, PIPX, PIPY)                CHL
* FINDS THE LOCATION IN PICBUF OF THE CHARACTER INDICATED BY THE PIPER. CHL
* HERE(LA) = PICBUF WORD AND HERE(UA) = BYTE (RT MOST=0, LF MOST=52B)   CHL
      CON(M6 =6B)                                                     CHL
 1ST  SLJ(N)        SLJ(L+6)      .                                   CHL
 2ST  ZRO(0)        ZRO(0)        . PICBUF.                           CHL
 3ST  ZRO(0)        ZRO(0)        . POSIT.                            CHL
 4ST  ZRO(0)        ZRO(0)        . HERE. IF HERE IS NEG.,  NOT FOUND.CHL
 5ST  ZRO(0)        ZRO(0)        . PIPX.                             CHL
 6ST  ZRO(0)        ZRO(0)        . PIPY.                             CHL
      LDA7(3ST)     SAU(5RNB)     .                                   CHL
      ENA(1SF)      SAU(1BRA)     .                                   CHL
      LDA(2ST)      SAU(3RNB)     . PREP.                             CHL
      ENI1(0)       ENI2(0)       . B1=BYTE  B2= WORD.                CHL
* READ NEXT BYTE.                                                     CHL
 1RNB IJP1(L+4)     ENI1(7)       . USE OLD WORD, JUMP.               CHL
```

```
 5RNB ISK2(N)         SLJ(3RNB)       . DICBUF EXAUSTED, JUMP.           CHL
 2RNB ENA(-1)         SLJ(1ST)        . NOT FOUND, EXIT.                 CHL
 3RNB LDA2(N)         STA(W)          . LOAD NEW WORD.                   CHL
      LDQ(W)          ENA(0)          .                                  CHL
      LLS(6)          STQ(W)          . BYTE IN A RJ.                    CHL
* MAIN BRANCH POINT.                                                     CHL
 1BRA SLJ(N)                          . MAIN BRANCH POINT.               CHL
* START FILE CHECK.                                                      CHL
 1SF  INA(-1)         AJP1(1RN3)      . NOT START FILE, JUMP.            CHL
      SLJ(3SS)                        . START FILE, GO TO DECODER.       CHL
* SKIP STRING.                                                           CHL
 1SS  ENA(2SS)                        . ROUTE STRING BELOW.              CHL
 2LDC SAU(1BRA)       SLJ(1RNB)       .                                  CHL
 2SS  INA(-77B)       AJP1(1RN3)      . NOT END OF STRING, JUMP.         CHL
 3SS  ENA(1DC)        SLJ(L-2)        . RESTORE ROUTE.                   CHL
* DECODER.                                                               CHL
 1DC  AJP(1SS)        INA(-20B)       . START STRING, JUMP.              CHL
      AJP(2PX)        INA(-1)         .                                  CHL
      AJP(2PY)        INA(-1)         .                                  CHL
      AJP(2MX)        INA(-1)         .                                  CHL
      AJP(2MY)        INA(-5)         .                                  CHL
      AJP(1SM)        INA(-1)         . SMALL                            CHL
      AJP(1ME)        INA(-1)         . MEDIUM.                          CHL
      AJP(1LR)        SLJ(1RNB)       . LARGE.                           CHL
 1SM  ENA(2)          SLJ(L+3)        .                                  CHL
 1ME  ENA(4)          SLJ(L+2)        .                                  CHL
 1LR  ENA(10B)                        .                                  CHL
      STA(SIZE)       SLJ(1RNB)       . SIZE = 1/2 OF TRUE SIZE.         CHL
 2PX  ENA(1PX)        SLJ(2LDC)       . ROUTE NEXT LOCATION BYTE.        CHL
 2PY  ENA(1PY)        SLJ(2LDC)       .                                  CHL
 2MX  ENA(1MX)        SLJ(2LDC)       .                                  CHL
 2MY  ENA(1MY)        SLJ(2LDC)       .                                  CHL
* LOCATION SEARCH.                                                       CHL
 1MX  ENQ(0)          LLS(2)          . -X.                              CHL
      SCM(1SEV)       SLJ(L+2)        .                                  CHL
 1PX  ENQ(0)          LLS(2)          . +X.                              CHL
      STA(XF)         SLJ(3SS)        . EXIT.                            CHL
 1MY  ENQ(0)          LLS(2)          . -Y. NOTE-Y POSITION ALWAYS FOLLOWSCHL
      SCM(1SEV)       SLJ(L+2)        .    X POSITION DUE TO 1LOC ROUTINE.CHL
 1PY  ENQ(0)          LLS(2)          . +Y.                              CHL
      STA(YF)         SUB7(6ST)       .                                  CHL
      AJP3(L+1)       SCM(1SEV)       . IF SIZE - ABS(YF - PIPY) IS      CHL
      ADD(SIZE)       AJP3(3SS)       .    NEG., WRONG LINE, JUMP        CHL
* CORRECT LINE FOUND. NOW FIND CHARACTER.                                CHL
 1CH  ENA(2CH)        SAU(1DC)        . REROUTE START STRING.            CHL
      SLJ(3SS)                        .                                  CHL
 2CH  ENA(1SS)        SAU(1DC)        . RESET START STRING.              CHL
      ENA(3CH)        SLJ(2LDC)       . ROUTE STRING BELOW.              CHL
 3CH  INA(-77B)       AJP(3SS)        . END STRING, REROUTE.             CHL
```

```
     LDA(XF)        SUB7(5ST)      .                                  CHL
     AJP3(L+1)      SCM(1SEV)      . IF SIZE - ABS(XF - PIPX) IS      CHL
     ADD(SIZE)      AJP2(1FND)     .   POS., RIGHT CHAR., JUMP.       CHL
     LDA(SIZE)      ADD(SIZE)      . INCREMENT XF.                    CHL
     RAD(XF)        SLJ(1RNB)      .                                  CHL
1FND ENA1(0)        MUI(M6)        . OUT PUT LOCATION                 CHL
     SAU7(4ST)                     .                                  CHL
     ENA2(0)        SAL7(4ST)      .                                  CHL
     LDA7(4ST)      SLJ(1ST)       . FOUND EXIT                       CHL
1SEV SEV7(77777B)   SEV7(77777B)   . ALL ONES.                        CHL
     END



     MACHINE FLASH                                                     FLA
1BG  SLJ(N),ENA(120B),STA(P),-SLJ4(1ON),                               FLA
    1-SLJ4(1OFF),-SLJ4(1L),          -SLJ4(1OFF),-SLJ4(1I),-SLJ4(1OFF),FLA
    2-SLJ4(1G),-SLJ4(1OFF),-SLJ4(1Z),-SLJ4(1OFF),-SLJ4(1T),-SLJ4(1OFF),FLA
    3          -SLJ4(1OFF),-SLJ4(1T),-SLJ4(1OFF),-SLJ4(1E),-SLJ4(1OFF),FLA
    4-SLJ4(1S),-SLJ4(1OFF),-SLJ4(1T),-SLJ4(1OFF),          -SLJ4(1OFF),FLA
    5ENA(20B),STA(P),ENI6(6).                                          FLA
2ST  EXF(77202B),SLJ4(1W),EXF(77203B),SLJ4(1W),IJP6(L-2),ENI6(6),    FLA
    1EXF(77204B),SLJ4(1W),EXF(77205B),SLJ4(1W),IJP6(L-2),ENI6(6),    FLA
    2EXF(77210B),SLJ4(1W),EXF(77211B),SLJ4(1W),IJP6(L-2),ENI6(6),    FLA
    3EXF(77220B),SLJ4(1W),EXF(77221B),SLJ4(1W),IJP6(L-2),ENI6(6),    FLA
    4EXF(77240B),SLJ4(1W),EXF(77241B),SLJ4(1W),IJP6(L-2),ENI6(6),    FLA
    5EXF(77340B),SLJ4(1W),EXF(77341B),SLJ4(1W),IJP6(L-2),ENI6(6),    FLA
    6EXF(77320B),SLJ4(1W),EXF(77321B),SLJ4(1W),IJP6(L-2),ENI6(6),    FLA
    7EXF(77310B),SLJ4(1W),EXF(77311B),SLJ4(1W),IJP6(L-2),ENI6(6),    FLA
    8EXF(77304B),SLJ4(1W),EXF(77305B),SLJ4(1W),IJP6(L-2),ENI6(6).    FLA
    0EXF(77302B),SLJ4(1W),EXF(77303B),SLJ4(1W),IJP6(L-2),ENI6(6),    FLA
    1EXF(77402B),SLJ4(1W),EXF(77403B),SLJ4(1W),IJP6(L-2),ENI6(6),    FLA
    2EXF(77404B),SLJ4(1W),EXF(77405B),SLJ4(1W),IJP6(L-2),ENI6(6),    FLA
    3EXF(77410B),SLJ4(1W),EXF(77411B),SLJ4(1W),IJP6(L-2),ENI6(6),    FLA
    4EXF(77420B),SLJ4(1W),EXF(77421B),SLJ4(1W),IJP6(L-2),ENI6(6),    FLA
    5EXF(77440B),SLJ4(1W),EXF(77441B),SLJ4(1W),IJP6(L-2),ENI6(6),    FLA
    6EXF(77540B),SLJ4(1W),EXF(77541B),SLJ4(1W),IJP6(L-2),ENI6(6),    FLA
    7EXF(77520B),SLJ4(1W),EXF(77521B),SLJ4(1W),IJP6(L-2),ENI6(6),    FLA
    8EXF(77510B),SLJ4(1W),EXF(77511B),SLJ4(1W),IJP6(L-2),ENI6(6).    FLA
    0EXF(77504B),SLJ4(1W),EXF(77505B),SLJ4(1W),IJP6(L-2),ENI6(6),    FLA
    1EXF(77502B),SLJ4(1W),EXF(77503B),SLJ4(1W),IJP6(L-2),ENI6(6),    FLA
    2EXF(77602B),SLJ4(1W),EXF(77603B),SLJ4(1W),IJP6(L-2),ENI6(6),    FLA
    3EXF(77604B),SLJ4(1W),EXF(77605B),SLJ4(1W),IJP6(L-2),ENI6(6),    FLA
    4EXF(77610B),SLJ4(1W),EXF(77611B),SLJ4(1W),IJP6(L-2),ENI6(6),    FLA
    5EXF(77620B),SLJ4(1W),EXF(77621B),SLJ4(1W),IJP6(L-2),ENI6(6),    FLA
    6EXF(77640B),SLJ4(1W),EXF(77641B),SLJ4(1W),IJP6(L-2),ENI6(6),    FLA
    7EXF(77740B),SLJ4(1W),EXF(77741B),SLJ4(1W),IJP6(L-2),ENI6(6),    FLA
```

```
     8EXF(77720B),SLJ4(1W),EXF(77721B),SLJ4(1W),IJP6(L-2),ENI6(6).          FLA
     0EXF(77710B),SLJ4(1W),EXF(77711B),SLJ4(1W),IJP6(L-2),ENI6(6),          FLA
     1EXF(77704B),SLJ4(1W),EXF(77705B),SLJ4(1W),IJP6(L-2),ENI6(6),          FLA
     2EXF(77702B),SLJ4(1W),EXF(77703B),SLJ4(1W),IJP6(L-2),ENI6(6),          FLA
     3-SLJ4(1ON),-SLJ4(1OFF),IJP6(L-2),ENI6(6),                             FLA
     4LDA(P),-AJP (1BG),ARS(1),STA(P),SLJ(2ST).                             FLA
1W    SLJ(N),LDA(P),AJP(L-1),ISK1( 4000B),INA(-1),SLJ(L-1).                 FLA
1OFF  SLJ(N),-SLJ4(1W),EXF(77203B),     EXF(77205B),     EXF(77211B),       FLA
      1 EXF(77221B),    EXF(77241B),    EXF(77303B),     EXF(77305B),       FLA
      2 EXF(77311B),    EXF(77321B),    EXF(77341B),     EXF(77403B),       FLA
      3 EXF(77405B),    EXF(77411B),    EXF(77421B),     EXF(77441B),       FLA
      4 EXF(77503B),    EXF(77505B),    EXF(77511B),     EXF(77521B),       FLA
      5 EXF(77541B),    EXF(77603B),    EXF(77605B),     EXF(77611B),       FLA
      6 EXF(77621B),    EXF(77641B),    EXF(77703B),     EXF(77705B),       FLA
      7 EXF(77711B),    EXF(77721B),    EXF(77741B),    -SLJ4(1W),          FLA
     8SLJ(1OFF).                                                            FLA
1L    SLJ(N),EXF(77302B),EXF(77304B),EXF(77310B),EXF(77320B),               FLA
     1EXF(77340B),EXF(77440B),EXF(77540B),                SLJ(1L).          FLA
1I    SLJ(N),EXF(77402B),EXF(77404B),EXF(77410B),EXF(77420B),               FLA
     1EXF(77440B),              SLJ(1I).                                    FLA
1T    SLJ(N),-SLJ4(1I),EXF(77502B),EXF(77302B),              SLJ(1T).       FLA
1E    SLJ(N),-SLJ4(1L),EXF(77402B),EXF(77502B),EXF(77410B),EXF(77510B),     FLA
     1              SLJ(1E).                                                FLA
1G    SLJ(N),-SLJ4(1E),EXF(77520B),SLJ(1G).                                 FLA
1S    SLJ(N),-SLJ4(1G),EXF(77321B),SLJ(1S).                                 FLA
1Z    SLJ(N),-SLJ4(1G),EXF(77504B),EXF(77403B),EXF(77441B),SLJ(1Z).         FLA
1ON   SLJ(N),-SLJ4(1Z),EXF(77202B),-SLJ4(1I),EXF(77204B),EXF(77210B),       FLA
     1EXF(77220B),EXF(77240B),EXF(77602B),EXF(77604B),EXF(77610B),          FLA
     2EXF(77620B),EXF(77640B),EXF(77702B),EXF(77704B),EXF(77710B),          FLA
     3EXF(77720B),EXF(77740B),SLJ(1ON).                                     FLA
      END
      END
```